SailPoint

Version 7.0 -2

Integration Guide

Table of Contents

IT Security Integration Modules 117

Chapter 15: SailPoint HP ArcSight Integration Module 119

Chapter 16: SailPoint Lieberman Integration 127

SAP GRC Integration Module 131

Chapter 17: SailPoint SAP GRC Integration Module 133

Appendix 151

Appendix A: Common Identity Management Integration Configuration 153

Chapter 1: Overview

The following topics are discussed in this chapter:

What is SailPoint IdentityIQ?

SailPoint IdentityIQ is an identity and access management solution for enterprise customers that delivers a wide variety of IAM processes-including automated access certifications, policy management, access request and provisioning, password management, and identity intelligence. Furthermore, IdentityIQ has a flexible connectivity model that simplifies the management of applications running in the datacenter or the cloud.

Compliance Manager: Compliance Manager automates access certifications, policy management, and audit reporting through a unified governance framework. This allows you to streamline compliance processes and improve the effectiveness of identity governance-all while lowering costs

Lifecycle Manager: Lifecycle Manager manages changes to access through user-friendly self-service request and password management interfaces and automated lifecycle events. It provides a flexible, scalable provisioning solution for addressing the constantly evolving access needs of your business in a way that's both efficient and compliant.

Unified Governance Platform: Governance Platform lays the foundation for effective IAM within the enterprise. It establishes a single framework that centralizes identity data, captures business policy, models roles, and takes a risk-based, proactive approach to managing users and resources. The Governance Platform also offers extensible analytics that transforms disparate, technical identity data into relevant business information. Additionally, robust resource connectivity is provided that allows organizations to directly connect to applications running in the datacenter or in the cloud.

A unified governance platform allows organizations to build a single preventive and detective control model that supports all identity business processes, across all applications-in the datacenter and the cloud. SailPoint IdentityIQ applies consistent governance across compliance, provisioning and access management processes, maximizing investment and eliminating the need to buy and integrate multiple products.

Integration Modules: The Integration Modules enables you to manage applications running in the datacenter or cloud through a flexible set of integration options. It directly connects to applications and provides out-of-box integration with other IT management tools, such as third-party provisioning, IT Service Management, Mobile Device Management or IT Security. This wide variety of connectivity options speeds deployment and reduces costs.

SailPoint Integration Guide Overview

SailPoint Integration Modules deliver extended value from standard IdentityIQ deployments. SailPoint is committed to providing design, configuration, troubleshooting and best practice information to deploy and maintain strategic integrations. SailPoint has modified the structure of this document to aid customers and partner deployments. The focus of this document is product configuration and integration. For more details on design, troubleshooting and deployment best practices, refer to the Connector and Integration Deployment Center in Compass, SailPoint's Online customer portal.

SailPoint Integration Guide Overview

This document provides a guide to the integration between the following products and IdentityIQ:

- **"Standard Deployment Integration Modules"**

    - Provisioning Integration Modules

        - SailPoint Oracle Identity Manager Provisioning Integration Module

        - SailPoint Sun Identity Manager Provisioning Integration Module

        - SailPoint Novell Identity Manager Provisioning Integration Module

        - SailPoint IBM Tivoli Provisioning Integration Module

    - Service Desk Integration

        - SailPoint BMC Remedy Service Desk Service Integration Module

        - SailPoint ServiceNow Service Integration Module

    - Service Catalog Integration

        - SailPoint BMC Remedy Service Desk Service Request Management Adapter

- **"Assisted Deployment Integration Modules"**

    - Provisioning Integration Modules
        - SailPoint Microsoft Forefront Identity Manager Provisioning Integration Module

    - Service Desk Integration
        - SailPoint HP Service Manager Service Integration Module

    - Service Catalog Integration

        - SailPoint ServiceNow Service Catalog Integration

    - Mobile Device Management Integration Modules
        - SailPoint AirWatch Mobile Device Management Integration Module

        - SailPoint MobileIron Mobile Device Management Integration Module

        - SailPoint Good Technology Mobile Device Management Integration Module

    - IT Security Integration Module

        - SailPoint HP ArcSight Integration Module

        - SailPoint Lieberman Integration Module

    - SAP GRC Integration Module

This document is intended for the above products and IdentityIQ System Administrators and assumes a high degree of technical knowledge.

Standard Deployment Integration Modules

This section contains information on the following sections:

- "Provisioning Integration Modules"
- "Service Management Integration Modules"

Standard Deployment Integration Modules:
Provisioning Integration Modules

IdentityIQ can be configured to integrate with provisioning providers to automate access management for your implementation. Provisioning providers can be configured to communicate user and account information and automatically add or revoke access.

This section contains information on the following:

> **Note:** **Many SailPoint customers have deployed the Integration Modules in this section. SailPoint still encourages deployment teams to obtain the latest troubleshooting and best practice information beyond what is contained in this document. For more specific information, refer to the Connector and Integration Deployment Center in Compass, SailPoint's Online customer portal.**

Chapter 2: SailPoint Oracle Identity Manager Provisioning Integration Module

The following topics are discussed in this chapter:

Overview

This chapter provides a guide to the integration between Oracle Identity Manager (OIM) and IdentityIQ. This chapter is intended for Oracle and IdentityIQ System Administrators and assumes a high degree of technical knowledge.

The integration is achieved by deploying a small web application in the application server that hosts OIM. IdentityIQ communicates with the web services contained in this application to read and write account information. Configuration of the OIM integration requires the username and password of the OIM administrator or another user with sufficient permissions.

Supported features

The Oracle Identity Manager Provisioning Integration Module supports the following functions:

- Account Management

    - Oracle Identity Manager user aggregation along with the connected child accounts and application

    - Create, Update, Delete

    - Enable, Disable, Unlock

    - Add/Remove Entitlements operations for Oracle Identity Manager connected child accounts

- User Management

    - Manages Oracle Identity Manager Users as Accounts

    - Create, Update, Delete

    - Enable, Disable, Unlock

    - Add/Remove Entitlements operations for Oracle Identity Manager Users

Supported platforms

SailPoint Oracle Identity Manager Provisioning Integration Module supports the following versions of Oracle Identity Manager:

- Oracle Identity Manager 11g R2
- Oracle Identity Manager 11g R1

Installing the OIM Integration Web Application

You must first deploy the OIM Integration Servlet web application to the application server hosting the OIM application. The `iiq.war` file for this web application is contained in the IdentityIQ distribution as `$INSTALLDIR/integration/OIM/iiqIntegration-OIM.jar` or in the distribution for an IdentityIQ patch in a `.jar` file named `Integration-oim-<version>.jar`.

The `iiqIntegration-OIM.jar` file contains `iiq.war` file. You can customize the `iiq.war` file in many ways before being deployed into the application server hosting OIM.

> **Note:** Ensure that if you are deploying web application as war file, it should be named as iiq.war. If you are deploying the web application from a directory, then directory should be named as iiq.

The following are the required customization steps:

1. Configure access to OIM by modifying `WEB-INF/classes/xellerate.properties` to set.

- **XL.HomeDir**: the full path to the directory where OIM is installed
- **userName**: the OIM administrator that has the appropriate permission to read and write user and account data
- **password**: the password for the OIM administrator

   For more information on the other properties that need to be set in xellerate.properties file, see "Properties that can be defined in xellerate.properties" on page 10.

2. Copy the required API implementation .jar files from the OIM installation into the `WEB-INF/lib` directory of the integration application:

- (*For Oracle 11g*) Copy `OIM_ORACLE_HOME/designconsole/lib/oimclient.jar`
- (*For Oracle 10g*)

   - Copy the following files from the OIM web lib directory to **iiq.war** only if **iiq.war** and OIM server are installed on separate WebLogic managed server:

      - `javagroups-all.jar`
      - `oscache.jar`
      - `wlXLSecurityProviders.jar`
      - `XIMDD.jar`
      - `XL10SecurityProviders.jar`

- Copy the following files from the OIM external lib directory
  - `xlAdapterUtilities.jar`
  - `xlAPI.jar`
  - `xlAttestation.jar`
  - `xlAuditor.jar`
  - `xlAuthentication.jar`
  - `xlBackOfficeBeans.jar`
  - `xlBackofficeClient.jar`
  - `xlCache.jar`
  - `xlCrypto.jar`
  - `xlDataObjectBeans.jar`
  - `xlDataObjects.jar`
  - `xlDDM.jar`
  - `xlGenConnector.jar`
  - `xlGenericUtils.jar`
  - `xliGCProviders.jar`
  - `xlInputPreprocessor.jar`
  - `xlInstaller.jar`
  - `xlLogger.jar`
  - `xlRemoteManager.jar`
  - `xlRequestPreview.jar`
  - `xlSampleApp.jar`
  - `xlScheduler.jar`
  - `xlUtils.jar`
  - `xlVO.jar`
  - `xlWebClient.jar`
  - `xlWSCustomClient.jar`

Testing the OIM Integration Web Application

Verify if the installation was successful using the following steps:

> **Note:** For each test URL throughout this document, change the host name and port to match your OIM Server instance.

1. From any browser enter the following URL:
   http://localhost:8080/iiq/resources/ping
   The following response is displayed:
   `OIM integration ready`
   Failure to get a ping response indicates a problem with the deployment of the servlet.

2. Verify the integration servlet can communicate with OIM by entering the following URL:
   http//localhost:8080/iiq/resources/users

   You should see a response containing the names of all OIM users. This might take a while to assemble depending on the number of users. To view details of a particular user, enter the following URL where `<OMIUSER>` is the name of a user in your OIM instance:

   http://localhost:8080/iiq/resources/user/<OIMUSER>

To see additional diagnostic information for of a particular user, enter the following URL where `<OMIUSER>` is the name of a user in your OIM instance:

http://localhost:8080/iiq/resources/debug/<OIMUSER>

If you are unable to request user information, there may be a problem with the credentials you entered in the `xellerate.properties` file. For more information, see "Properties that can be defined in xellerate.properties" on page 10.

Properties that can be defined in xellerate.properties

1. Add a ManagedResource definition in the ManagedResource list for an each OIM resource. For each resource, define a property prefix by adding a property whose name is the prefix and whose value is the OIM resource name.

   For example:
   AD=AD User

   Oracle=Oracle DB User

   This declares that any property that begins with ERP is related to the OIM resource named ERP Central Component.

2. For each ManagedResource, define the account attribute that represents the unique account identifier. The names used here must be the resource names used by OIM. The identityAttribute must have the internal form field name containing the account identifier. Use the OIM Design Console application to find the process form for each resource and view the field names. The example below gives two typical names, one used by the connector for Oracle database users and the other for the Active Directory connector.
   ```
   AD.id=UD_ADUSER_UID
   Oracle.id=UD_DB_ORA_U_USERNAME
   ```

3. Define the names of the child forms that support multiple attributes. The value is a CSV of the internal child form names:
   ```
   AD.childForms=UD_ADUSRC
   Oracle.childForms=UD_DB_ORA_R
   ```

   In this example UD_ADUSRC is the internal name for the child form AD User Group Details and UD_DB_ORA_R is the internal name for the child form DBUM Grant/Revoke Roles.

4. Each child form name in the Oracle.childForms property there is another property whose value is a CSV of the child form fields to return and the order in which they will appear in IdentityIQ.
   ```
   Oracle.UD_DB_ORA_R=UD_DB_ORA_R_ROLE,UD_DB_ORA_R_ADMIN_OPTION
   ```

   In the previous example, we will return two fields from the child form UD_DB_ORA_R. The first field has the Role name and the second has the Role Admin option.

5. Following is the configuration for resource with child forms: ERP Central Component:
   ```
   ERP=ERP Central Component
   ERP.id=UD_ECC_USER_ID
   ERP.childForms=UD_ECC_PRO,UD_ECCRL
   ERP.UD_ECC_PRO=UD_ECC_PRO_SYSTEMNAME,UD_ECC_PRO_USERPROFILE
   ERP.UD_ECCRL=UD_ECCRL_SYSTEMNAME,UD_ECCRL_USERROLE
   ```

Note: Before IdentityIQ 6.0 there was a parameter in **xellerate.properties** file as **oldChildFormNames** which was used for the resources who have only one field in the childform, for example, Active Directory resource. For **IdentityIQ** version 6.0 onwards, the value must be set to true if the user wants to support **oldChildFormNames** where field returned would be form name + field name (For example, **UD_ADUSRC:UD_ADUSRC_GROUPNAME** field in Active directory).

6. To aggregate all the active and disabled OIM users in IdentityIQ, add a new parameter **OIM_USER_TYPE** in `xelerate.properties` file with the value as **ALL**. If **OIM_USER_TYPE** parameter is removed from the `xelerate.properties` file then only the active OIM users will be aggregated. By default only active OIM user are aggregated.

Configuration for OIM application

Perform the following steps to create an IdentityIQ application for OIM:

1. Navigate to the IdentityIQ **Define=>Application** page.

2. Create a new application of type **Oracle Identity Manager**.

3. On the Attributes tab, enter the Oracle Identity Manager Host and Oracle Identity Manager Port.

4. Click **Test Connection** to verify the connection to OIM.

    Note: **You can make use of the "OIM Application creator" task to discover all the resources present in OIM environment. The input for this task would be an newly created application of type "Oracle Identity Manager" and executing this task would result in the creation of all multiplexed resources.**

Testing the OIM Integration Client

While any IdentityIQ feature that generates a provisioning request such as a certification remediation, a role assignment, or a Lifecycle Manager request can be used to test the integration, it is sometimes useful to test at the provisioning layer using the IdentityIQ integration console.

Launch the console by using the IdentityIQ script in the `INSTALLDIR/WEB-INF/bin` directory of the IdentityIQ installation to run iiq integration.

From the console command prompt, use the **list** command to display the names of all Application objects created in the system. Using the example in the previous section, verify an Application object of type **Oracle Identity Manager** exits.

Use the following command:

`use OIMApplicationName`

Use the **ping** command to initiate a test connection message with OIM. A successful connection will return the following message:

`Response: Connection test successful`

If any problem occurs in the communication of this application with the OIM Integration Web Application, troubleshoot this application by viewing the application server logs for both the IdentityIQ and OIM application servers. You can enable log4j tracing on both sides by using the following:

`log4j.logger.sailpoint.integration=debug`

`log4j.logger.sailpoint.connector=debug`

This lets you see if the requests are transmitting over the network, and how they are processed.

If the OIM servlet is deployed on Weblogic 11g, tracing can be enabled on it by adding an entry to the logging file on the Weblogic server. Following is the logging file:

`<DOMAIN_HOME>/config/fmwconfig/servers/oim_server1/logging.xml`

Following is the entry that needs to be added:

```
<logger name="SailPoint.integration.oim" level="TRACE:32"/>
```

For more information on enabling system logging in OIM is included in the *Oracle Identity Manager Administrator Guide*.

Aggregating from OIM

To aggregate OIM users and resource accounts, create and execute an IdentityIQ Account Aggregation task. Include the OIM application in the applications to scan list.

When the aggregation is complete from the OIM application, a new application is created for every resource in OIM. The application schema includes attributes seen in the resource accounts. All users in OIM have an account created that is associated with the OIM application and includes all of the standard and extended user attributes of those users. Additionally, all of the resource accounts are aggregated and associated with the newly created applications.

Once the initial aggregation from the OIM application is completed, you can aggregate from it again to read in information for all managed systems.

> **Note:** **You can make use of the "OIM Application Creator task" to discover all of the Resources present in the OIM environment. The input for this task is an application of type Oracle Identity Manager. Executing this task results in the creation of all multiplexed Resource applications.**

Known/Open issues

Following is the known/open issue of Oracle Identity Manager:

- You cannot perform provisioning operations simultaneously on the OIM server from IdentityIQ and the OIM console. This is a class loading issue observed with OIM 11g, after deploying iiq servlet(iiq.war) on Weblogic OIM Managed Server.

  Workaround for this issue: Create another, empty WLS(Weblogic)Managed server next to the OIM Managed Server and only deploy the IIQ Servlet. Also, update the **Xellerate.properties** file by un-commenting the attribute **java.naming.provider.url**. This Url needs the host name of the host where OIM managed server is deployed and the listening port of the OIM managed server.

- Create OIM user and Update OIM user operations are not working with Oracle Identity Manager 11g R2.

Chapter 3: SailPoint Sun Identity Manager Provisioning Integration Module

Introduction

The integration between IdentityIQ and Sun Identity Manager is very straightforward and requires little configuration on either side to start the initial communication. This document covers the important points for configuring an integration between these products.

Supported features

The Sun Identity Manager Provisioning Integration Module provides the ability to provision Sun Identity Manager users, Target Application accounts, groups and entitlements from IdentityIQ.

The Provisioning Integration Module supports the following functions:

- User Management

    - Manages Sun Identity Manager Users as Accounts

    - Aggregating Users

    - Create, Update, Delete

    - Enable, Disable, Unlock, Reset Password

    - Add/Remove User Entitlements

- Target Application Accounts Management

    - Manages Target Application Accounts as Accounts

    - Target Accounts are aggregated as part of Sun IdM User aggregation

    - Create, Update, Delete

    - Enable, Disable, Reset Password

    - Add/Remove Account Entitlements

- Group Management

  - Manages Sun Identity Manager Groups as Account-Groups in IdentityIQ

  - Groups are aggregated as part of user aggregation task

Supported platforms

SailPoint Sun Identity Manager Provisioning Integration Module supports the following version of Oracle Waveset:

- Oracle Waveset 8.1.1

General configuration

IdentityIQ communicates with Sun Identity Manager using a Service Provisioning Markup Language (SPML) interface. To enable this communication, the SPML client library, `openspml.jar`, must be copied from the Sun IdM installation into the `INSTALLDIR/WEB-INF/lib` directory of the IdentityIQ installation and then the IdentityIQ application should be restarted. A system error will occur in IdentityIQ when defining a Sun Identity Manager application before this configuration step is completed.

Configuration for Aggregation

Configuring IdentityIQ to aggregate accounts from Sun Identity Manager is accomplished by creating an IdentityIQ application as outlined in the following steps:

Configure SPML in Sun Identity Manager

1. If you do not have a Configuration:SPML object in Sun Identity Manager, import **sample/spml.xml** from the Sun Identity Manager installation.

   Note: **SPML v1 requests are used only for Aggregation and Provisioning.**

2. Add the following object into the SPML classes list:
   ```
           <Object name='IIQUserView'>
             <Attribute name='type' value='User'/>

             <!-- keep the default form quiet -->
             <Attribute name='form'       value='view'/>
             <Attribute name='viewForm'   value='Empty'/>
             <Attribute name='identifier' value='waveset.accountId'/>
             <Attribute name='default'    value='true'/>

        <!-- May need this if the install ordinarily turns on the meta-view -->
             <Attribute name='viewOptions'>
               <Object>
                 <Attribute name='ApplyMetaView' value='false'/>
               </Object>
             </Attribute>
   ```

```
<!-- This is important for ModifyRequests to remove selected elements.
     We'll have to put something here for every multi-valued attribute
  in every managed resource.  Tedious but at this point necessary.
   <Attribute name='multiValuedAttributes'>
     <List>
       <String>IDM/IdentityIQ Integration Demo::groups</String>
       <String>accounts[IDM/IdentityIQ Integration Demo].groups</String>
     </List>
   </Attribute>
   -->
</Object>
```

Define **IdentityIQ** Application

1. The object name (in this case IIQUserView) should match the Native Object Type of the account schema IdentityIQ application that represents Sun Identity Manager.

2. When creating the Sun Identity Manager application in IdentityIQ, follow the default pattern for the rpcRouterURL to point to the Sun Identity Manager system.

To aggregate Sun Identity Manager users and resource accounts, create and execute an IdentityIQ Account Aggregation task for the Sun Identity Manager application.

When the aggregation task is complete, a new application is created for every managed resource in Sun Identity Manager. The application schema includes attributes from the resource accounts. All users in Sun Identity Manager have an account created that is associated with the Sun Identity Manager application and any administrative capabilities and assigned profiles are included as attributes or direct permissions of those accounts.

Once the initial aggregation from the Sun Identity Manager application is completed, you can aggregate from it again to read in information for all of the resource systems.

Configuration for Provisioning

No additional configuration is needed to enable provisioning to Sun Identity Manager. The Sun Identity Manager PIM is implemented using the latest IdentityIQ connector interface that provides read/write capability. If you wish to provision accounts for Sun Identity Manager applications using another method, then you must remove the **PROVISIONING** keyword from the **featureString** in the Sun Identity Manager application in IdentityIQ.

Chapter 4: SailPoint Novell Identity Manager Provisioning Integration Module

The following topics are discussed in this chapter:

Overview

This chapter provides a guide to the Novell® Identity Manager and SailPoint integration and configuration for your enterprise. This chapter is intended for Novell and IdentityIQ System Administrators and assumes a high degree of technical knowledge of these systems.

Supported features

The Novell Identity Manager Provisioning Integration Module provides the ability to provision Novell Identity Manager users, Target Application accounts and entitlements from IdentityIQ.

The Provisioning Integration Module supports the following functions:

- User Management

  - Manages Novell Identity Manager Users as Accounts

  - Aggregating Users

  - Create, Update, Delete

  - Enable, Disable

- Target Application Accounts Management

  - Manages Target Application Accounts as Accounts

  - Target Accounts are aggregated as part of Novell Identity Manager User aggregation

  - Create, Update, Delete

  - Enable, Disable

  - Add/Remove Account Entitlements

- Group Management

  - Manages Novell Identity Manager Groups as Account-Groups

  - Aggregating Groups
    Note: **While aggregating Novell IdM Users/Groups, the connected target system Accounts/Groups are also aggregated.**

  Note: **Before you can use the provisioning feature of the connector, all PRDs must be defined and configured on Novell side. For more information about the PRD's, see "IdentityIQ Provisioning workflows" on page 20 section.**

Supported platforms

SailPoint Novell Identity Manager Provisioning Integration Module supports the following version of Novell Identity Manager:

- Novell Identity Manager 4.0

Pre-requisites

It is mandatory to have the user application module installed on the Novell Identity Manager.

Configuring Novell Identity Manager for IdentityIQ Integration

The integration has dependencies on the following Novell Identity Manager components:

- Connected Systems Drivers
- Role Based Entitlements (RBEs)
- User Service Driver
- IdentityIQ Provisioning Workflows
- Enabling the Web Service

Connected systems

The DirXML drivers, connected to the Identity Vault from which the data is aggregated, have to be mapped to IdentityIQ Applications. This is done using the Novell Application Generator task (see below).

Role Base Entitlements (RBEs)

Role Based Entitlements (RBEs) need to be defined in the Identity Vault before automatic remediation from IdentityIQ can occur. Policies can be setup in the vault to add and remove account membership, group membership, etc. when these RBEs are granted or revoked. When accounts are aggregated into IdentityIQ, these RBEs show up as IdentityIQ entitlements.

User Service Driver configuration

The user service driver provides the provisioning and directory abstraction layer (VDX) web service that IdentityIQ uses to communicate with Novell Identity Manager to perform remediation and aggregation respectively. It is assumed that the user service driver is the Entitlement granting authority.

> **Note:** **A conflict occurs if you have an Entitlement Service Driver for the same driverset.**

IdentityIQ Directory entities

Directory entities must be defined for the User Service driver that are used by IdentityIQ when it invokes the VDX web service. This entity is defined for the User Service driver under Directory Abstraction Layer.

For Account aggregation, entity must be created with the details mentioned in the following table:

Task	Entity Key	Object Class	Attributes
For Novell Application Creator Task	dirXML-Driver	DirXML-Driver	CN (allow search access to users), Object Class (allow search access to users)
	For Novell 4.*x* the following entity is also required		
	DirXML-Entitlement	DirXML-Entitlement	CN (allow search access to users), Object Class (allow search access to users). Provide value for the Search Container
For Account Aggregation	user	User	CN, Department, Direct Reports, DirXML-Accounts, DirXML-Associations, DirXML-EntitlementRef, Email, First Name, Group, Hidden attribute List, Last Name (allow search access to users), Login Disabled, Manager, Preferred Notification, Query List, Region, Telephone Number, Title
For Account Group Aggregation	group	Group	CN, Description (allow search access to users), DirXML-Associations, DirXML-EntitlementRef, Members

> **Note:** - For entity **DirXML-Driver**, the attributes CN and objectClass are mandatory and they should be marked as Searchable.
> - For **user** and **group** entities, the attribute keys should be in sync with account and group schema attribute names in IdentityIQ.
> - For user entity, the **LastName** attribute is mandatory and it should be marked as Searchable.
> - For group entity, the **Description** attribute is mandatory and it should be marked as Searchable.

Save these entities and import into the identity vault. Note the key of these entities as it is used in IdentityIQ.

IdentityIQ Provisioning workflows

Provisioning workflows must be defined for the User Service driver that are used by IdentityIQ when it invokes the provisioning web service. This workflow is defined for the User Service driver under Provisioning Request Definitions. You can find sample workflows (PRD) under integration/nidm/exampleWorkflows. Import them under Provisioning Request Definitions. After importing add a valid object in Trustee rights. Sample workflows (PRD) for the corresponding operations are as described in the following table:

PRD	Operation	Form fields
SPCreateModifyUser.xml	Create and Update Vault User	FirstName, LastName, Title, Location, Department, Email, loginDisabled, manager, group
SPDeleteUser.xml	Delete Vault User	Form fields are not required for this workflow.
SPDisableEnableUser.xml	Enable and Disable Vault User	loginDisabled
SPEntitlements.xml	Request Access for Vault User	entitlementReceiver, entitlementDn, entitlementAction, entitlementParameter

> **Note:** Additional workflow steps and logic might be required based on your business needs.
> - **For Create and update user workflow, the field names and number of fields in provisioning policy on IdentityIQ side should be same as that of the input field names of workflow.**
> - **For Enable/Disable vault user workflow and Request access workflow (that is, Entitlement), the names of input parameters of workflow should be same as given in above table. Also all mandatory fields on Novell side should be marked as "required" in IdentityIQ provisioning policy.**

To define a workflow with the minimum required steps, perform the following:

1. Create a provisioning request definition of type Accounts/Entitlements.

2. Define a simple workflow with three steps, Star, Entity/Entitlement and Finish.

3. Define a form with fields as shown in the Sample PRD below:

4. Verify in the Entity/Entitlement step of the workflow that the Source expressions are defined correctly.
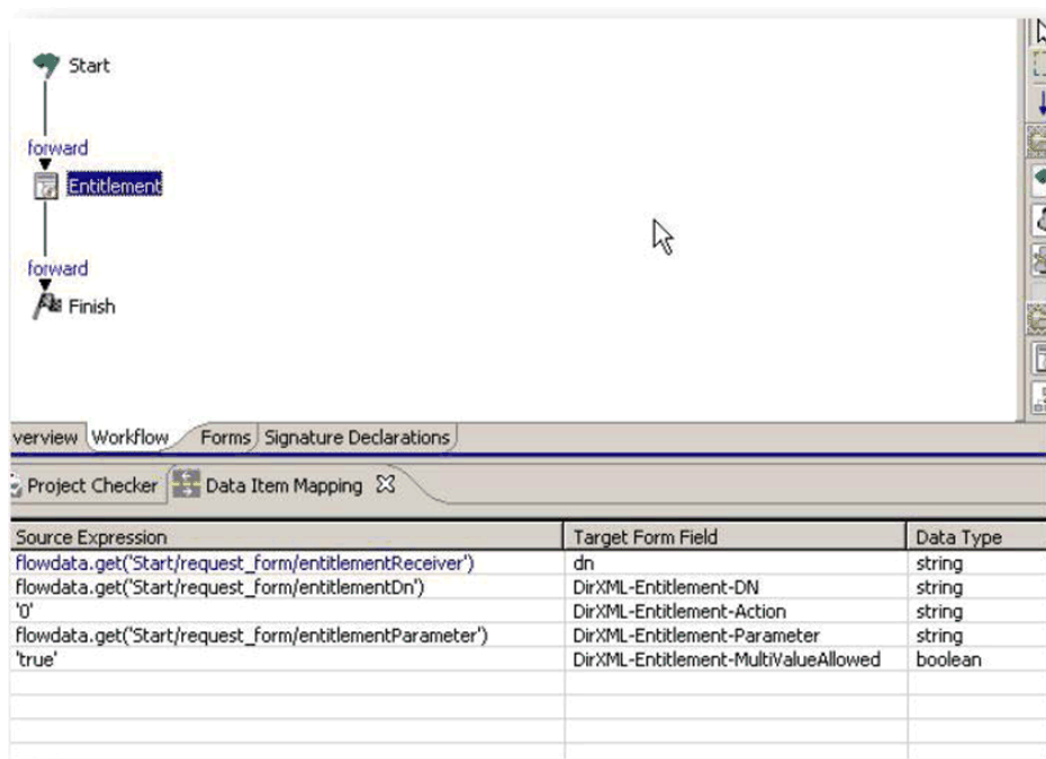
5. Save this workflow and import into the identity vault. Note the dn of this workflow as it is used in defining the Novell Application attributes later in the process.

Enabling the Web Service

The integration uses the following Web Service endpoints provided:

- Directory Abstraction Layer (VDX) Web Service for aggregation
- Provisioning Web Service for remediation

By default, the endpoints are disabled. To provide necessary credentials to the user, see the following sections in the *Novell User Application: Administration Guide:*

- Removing Administrator Credential Restrictions
- Enabling the Test Page

Configuring IdentityIQ for Novell Identity Manager Integration

This section describes the procedure for Novell Identity Manager Integration.

Create Novell IDM application in IdentityIQ

Create a new IdentityIQ application of type Novell Identity Manager. Enter the following required Novell Identity Manager information:

- **Novell Version**: Version of the Novell Identity Manager. The version will accept a value of either 3 or 4. In case any other value is provided, the default value 3 is used.
- **User Application URL**: URL of the User Application, in the following format:

  http://*applicationServerHost:port/applicationContext*
- **Username**: User Application Administrator who has the rights to administer Web Services such as Provisioning using workflow tasks and Directory Abstraction Layer (VDX)
- **Password**: Password of the user specified above.

Define Provisioning Request Definition for Novell application

Enter the dn for the workflow created in as application attributes:

1. To Create Vault User: **Create Vault User**

2. To Delete Vault User: **Delete Vault User**

3. To Disable Vault User: **Disable Vault User**

4. To Enable Vault User: **Enable Vault User**

5. To Modify Vault User: **Modify Vault User**

6. To all the operations on Resource Accounts on Applications (Entitlements). This includes Create, Delete, Disable, Enable Resource Accounts or Add/Remove Entitlements: Entitlements

   Note: **Account Entitlement for any driver should contain "account" substring in it.**

Run the Novell Application Creator Task

This task creates an IdentityIQ application for each connected systems driver in Novell® Identity Manager.

1. Select the Novell application created in the previous section.

2. Click **Fetch Novell Applications**. This shows all the connected systems. Select the ones with which you want IdentityIQ to integrate.

3. Click **Save and Execute**.

4. It will generate child applications for all drivers selected in IdentityIQ.

To fetch Account Status of Account Entitlement, add the following entry in Application's attribute, from the Application debug page:

```
<entry key="statusAttribute" value="loginDisabled"/>
```

In the above entry, **loginDisabled** represents the Status of Accounts on that driver.

Additional steps for Novell 4.0

Following manual configuration changes are required on IdentityIQ as there are number of configuration changes between Novell versions 3.x and 4.x:

1. After defining Novell application, change few attribute names in account and group schema for example, cN to CN, dirXML-Associations to DirXML-Associations. Check attribute names of Novell entity.

2. For generated child applications, add the following attributes in account schema:
```
<AttributeDefinition multi="true" name="Account" type="string"/>
<AttributeDefinition entitlement="true" managed="true" multi="true" name="Group"
type="string"/>
```
Set the Group attribute as group attribute of account schema.

3. Parent Novell application definition in IdentityIQ contains mapping between attribute names in version 4.0 which are used in application internally.
```
<entry key="4.0">
    <value>
        <Map>
            <entry key="cN" value="CN"/>
            <entry key="dirXML-Associations" value="DirXML-Associations"/>
            <entry key="dirXML-EntitlementRef"
value="DirXML-EntitlementRef"/>
            <entry key="LastName" value="LastName"/>
            <entry key="Description" value="Description"/>
            <entry key="objectClass" value="ObjectClass"/>
            <entry key="gUID" value="GUID"/>
            <entry key="loginDisabled" value="LoginDisabled"/>
            <entry key="dirXML-Driver" value="DirXML-Driver"/>
            <entry key="entitlementParameter">
                        <value>
                            <String>{"ID":"nidm.com"}</String>
                        </value>
            </entry>
        </Map>
    </value>
</entry>
```

4. By default, from IdentityIQ, while sending **User Account Entitlement** or **Group Entitlement**, parameter for-
mat is assumed as **Identity Manager 4**. If parameter format is configured as **legacy**, then add the following
properties at child application for which parameter format is to be configured:

```
<entry key="accountEntitlementParamFormat" value="legacy"/>

<entry key="groupEntitlementParamFormat" value="legacy"/>
```

If these properties are absent in application, then IdentityIQ will send parameter format in Identity Manager
4 format.

For Novell version 4.0, **entitlementParameter** indicates name of domain name of Novell setup. Change the value
from `nidm.com` to the domain name on which Novell is installed.

Aggregation from Novell PIM

To aggregate Novell users and resource accounts, create and execute an IdentityIQ Account Aggregation task.
Include the Novell application in the applications to scan list.

Once aggregation is complete, all users in Novell have an account created that is associated with the Novell
application and includes all of attributes defined in Novell application account schema. Additionally, all of the
resource accounts are aggregated and associated with the child applications.

Note: Before running the aggregation task, ensure "Run the Novell Application Creator Task" on
page 22 has been executed.

Chapter 5: SailPoint IBM Security Provisioning Integration Module

The following topics are discussed in this chapter:

Overview

This chapter is designed to provide the necessary procedures, configuration steps, and general product guidelines to successfully integrate IBM Security® Identity Manager (ISIM) into your SailPoint production environment.

This chapter is intended for ISIM and IdentityIQ System Administrators and assumes a high degree of technical knowledge of these systems.

Supported features

The IBM Security Identity Manager Provisioning Integration Module provides the ability to provision Target Application accounts from IdentityIQ.

The Security Provisioning Integration Module supports the following functions:

- User Management

    - Manages IBM Security Identity Manager Users as Accounts

    - Aggregating Users

- Target Application Accounts Management

    - Manages Target Application Accounts as Accounts

    - Aggregating Target Accounts directly

    - Create, Update, Delete

    - Enable, Disable, Reset Password

Supported platforms

SailPoint IBM Security Provisioning Integration Module supports the following versions of Security Identity Manager:

- IBM Security Identity Manager 7.0
- IBM Security Identity Manager 6.0
- IBM Security Identity Manager 5.1
- IBM WebSphere Application Server 5.1, 6.1 and 7.0

General configuration

The installation steps for ISIM integrations vary based on the functions you wish to perform. IdentityIQ in conjunction with ISIM allows the following functionality:

- Aggregation
- Provisioning Entitlements in ISIM

Configuration for Aggregation

Aggregating from IBM Security Identity Manager involves configuring the ISIM application settings within the IdentityIQ user interface.

ISIM has two types of objects that can be aggregated; people and accounts. IdentityIQ refers to these as identities and accounts (or links). To aggregate from ISIM, perform the following:

1. **Create An ISIM Application**: Create a new application using the IBM Security Identity Manager connector and fill in the required parameters following the steps provided in the IdentityIQ User's Guide.
   Use the tenant DN search base. For example,
   `erglobalid=00000000000000000000,ou=example,dc=com`

   Leave the search filter blank. This is auto-generated correctly during aggregation. This application is used to aggregate ISIM person objects.

2. **Setup Correlation Attribute**: Create an identity attribute that is sourced from the `erglobalid` on the ISIM application and mark it as searchable. This is used to correlate ISIM accounts to this identity.

3. **Create ISIM Account Applications**: Run the ISIM Application Creator task to inspect ISIM and retrieve information about the ISIM services (applications). This task auto-generates an application for each service defined in ISIM.

4. **Setup Correlation on the ISIM Account Applications**: Set the correlation rule on the generated applications to Correlation - ISIM Account. This correlates the account to the identity using the `erglobalid`. If the rule is not listed by default, import it from the `$ISIM_INTEGRATION_PACKAGE/resources/ISIM-Account-CorrelationRule.xml` location.

5. **Aggregate**: Run aggregation for the ISIM application first and then for each ISIM account application.

Configuration for Provisioning

Provisioning entitlements and role assignments in ISIM requires the installation of IdentityIQ's ISIM integration web application in WebSphere with ISIM. This process varies slightly depending on the version of WebSphere.

IdentityIQ roles are queued and pushed in ISIM on a schedule. This is accomplished by using the Synchronize Roles task.

1. **Prepare the WAR:** The `iiqIntegration-ISIM.war` file contains a properties file named `ISIM.properties` with information about how to connect using ISIM. In order to execute, this must be edited to include appropriate information about the ISIM installation. Additionally, the **.war** file does not include any of the required **ISIM.jar** files since these can change depending on the version and fixpack level of ISIM. These need to be copied out of the ISIM lib directory and added to the **.war** file.
    a. Expand the `iiqIntegration-ISIM.war` file in a temporary directory.
    b. Edit the `WEB-INF/classes/isim.properties` file and change the properties match your environment. Save the file with your changes. The following can be changed:

        - **PLATFORM_URL:** URL to use to communicate with ISIM.

        - **PLATFORM_PRINCIPAL:** Principal to connect as.

        - **PLATFORM_CREDENTIALS:** Password of the principal. Encrypting password is supported.

        - **TENANT_DN:** The root DN of the ISIM tenant.
    c. Copy the required `ISIM.jar` files into the lib directory. These **.jar** files are located in the deployed ISIM ear directory.

    - (*For ISIM 4.6*): Example ISIM ear directory: $WAS_HOME/installedApps/isim/enRole.ear

        Following are the required files:

        - `api_ejb.jar`

        - `isim_api.jar`

        - `isim_server.jar`

        - `jlog.jar`

    - (*For ISIM 5.0 and 5.1*): Example ISIM ear directory:
        `$WAS_HOME/profiles/<app server>/installedApps/<cell>/ISIM.ear`
        Following are the required files:

        - `api_ejb.jar`

        - `isim_api.jar`

        - `isim_common.jar`

        - `isim_server_api.jar`

        - `jlog.jar`

    d. Update the `iiqIntegration-ISIM.war` file to include the updated `isim.properties` and required `ISIM.jar` files.

        For example,
        ```
        jar uvf iiqIntegration-ISIM.war WEB-INF/classes/isim.properties \
        WEB-INF/lib/api_ejb.jar WEB-INF/lib/isim_api.jar \
        ```

```
WEB-INF/lib/isim_common.jar WEB-INF/lib/isim_server_api.jar \
WEB-INF/lib/jlog.jar
```

2. Install the IdentityIQ **ISIM Integration Web Application**: In the WebSphere Administrative Console, navigate to Enterprise Applications and select **Install.**

   a. Select **iiqIntegration-ISIM.war** as the application to install and type **iiqisim** as the context root.

   b. Continue through the rest of the installation wizard accepting the defaults.

   c. When completed, click **Save** to save the changes to the master configuration.

3. **Setup the Integration Config**: The **IntegrationConfig** object holds information about how to connect IdentityIQ to ISIM and all of the configuration requirements for various functions. ISIM supports dual role push mode, which means that both detectable and assignable roles can be used. An example can be found in the ISIM integration folder within your IdentityIQ installation directory in the `$INSTALLDIR/integration/ISIM/server/ resources/exampleIntegration.xml` directory.
   The main properties that need to be set are:

   - **executor**: sailpoint.integration.isim.ISIMIntegrationExecutor

   - **ApplicationRef**: The reference to the ISIM application

   - **Attributes**=> **URL**: The URL to the IIQ web service on the ISIM server. For example,
     https://myisim.example.com:9080/iiqisim/resources
     Note: **It is highly recommended that you use SSL when transmitting sensitive electronic information.**

   - **Attributes**=> **username**: ISIM user's credentials used for basic HTTP authentication.

   - **Attributes**=> **password**: ISIM user's password used for basic HTTP authentication.

   - **ManagedResources map**: Mappings of local IdentityIQ applications to ISIM services, including mappings of local IdentityIQ attribute names to ISIM service attribute names.
   For more information, see Appendix: A: Common Identity Management Integration Configuration**.**

4. **Verify**: Be certain that the integration has been installed correctly by using the ping command in the integration console. If successful, this should respond and list version information about the ISIM jar files that were put into the **iiqIntegration-ISIM.war** file. Compare this version information against the version of the ISIM server to ensure correct operation.

5. **Role Requests**: Set the roleSyncStyle to **dual** in the IntegrationConfig file as follows:
   ```
   <IntegrationConfig executor="sailpoint.integration.isim.ISIMIntegrationExecutor"
   name="ISIM 5.1 Integration" roleSyncStyle="dual">
   ```
   Other than this, the role should be assignable (for example, a business role) and the name has to match the name of the role in ISIM.

Standard Deployment Integration Modules:
Service Management Integration Modules

IdentityIQ supports Service Management Systems for Service Desk Integration and Service Catalog Integration.

Service Desk Integration

Creates tickets in Helpdesk systems based on compliance and provisioning actions taken in IdentityIQ (For example, new account request).

Service Catalog Integration

Enables enterprises to include LCM requests (for example, access request, password change) in the service catalog of the Service Request Management system.

Note: **Many SailPoint customers have deployed the Integration Modules in this section. SailPoint still encourages deployment teams to obtain the latest troubleshooting and best practice information beyond what is contained in this document. For more specific information, refer to the Connector and Integration Deployment Center in Compass, SailPoint's Online customer portal.**

Chapter 6: SailPoint BMC Remedy Service Desk Service Integration Module

The following topics are discussed in this chapter:

Overview

The integration between SailPoint and BMC Remedy Service Desk enables customers to create incidents and change requests in BMC Remedy Service Desk for the configured operations (for example, Change Password, Request Entitlement and so on) for the configured application. The seamless integration of SailPoint and BMC Remedy Service Desk Service Integration Module eliminates the need to build and maintain a custom integration, and speeds time-to-deployment.

Supported features

BMC Remedy Service Integration Module supports the following features:
- creating ticket for all provisioning operations that can be performed on Target Application accounts
- getting the status of the created tickets
- creating multiple tickets in Remedy System via IdentityIQ

Supported platforms

SailPoint BMC Remedy Service Desk Service Integration Module supports the following versions of BMC Remedy AR System:
- BMC Remedy AR System 9.0
- BMC Remedy AR System 8.1.00

Pre-requisites

- BMC Remedy Change Management Application must be installed
- Ensure that the following softwares are operating correctly:
    - BMC Remedy AR System
    - BMC Remedy Change Management Application

Basic configuration

The integrated solution speeds the detection and remediation of identity management issues that increase the risk of compliance violations or security breaches, such as orphaned accounts, policy violations, and inappropriate access privileges. Organizations can take advantage of a centralized approach spanning thousands of users and hundreds of resources to strengthen IT controls and provide proof of compliance to auditors and executive management. The seamless integration of SailPoint and BMC Remedy eliminates the need to build and maintain a custom integration, and speeds time-to-deployment.

For any IT resources managed by BMC Remedy Service Desk, IdentityIQ automatically creates a trouble ticket within Remedy Service Desk, passing along all relevant identity data and reviewer comments to populate the ticket.

To ensure revocation requests get delivered and implemented, IdentityIQ manages all remediation and revocation requests within a guaranteed delivery model.

To determine the status of user accounts, IdentityIQ performs closed-loop audits on remediation requests and compares the actual state of user privileges with the original change request. If the request is still open, an alert will be sent to the reviewer for prompt action and closure.

The integration itself has been designed to be quick to install and easy to use. It makes use of Web Services via the Remedy Mid Tier to broker communications between the SailPoint server and the AR System server. On the backside of a user recertification, policy remediation action or access request action, the IdentityIQ server will direct provisioning and service desk requests to the configured implementers. Based on the IntegrationConfig configured for each target application, service desk request are issues to a given remediation/implementation point. Once the IntegrationConfig for Remedy has been loaded into the IdentityIQ server, all change/remediation actions result in the creation of new service desk request.

At the completion of the change control cycle within IdentityIQ, an "Open Ticket" request is made over the appropriate SOAP channel to the Mid Tier. From here change request tickets are opened and the new ticket number is returned to IdentityIQ. The schema for the service request is defined in the IntegrationConfig and allows for the flexibility to transfer complete details on the service desk request. The default settings will create a basic ticket as shown in the following figure (Figure 1—Change request).
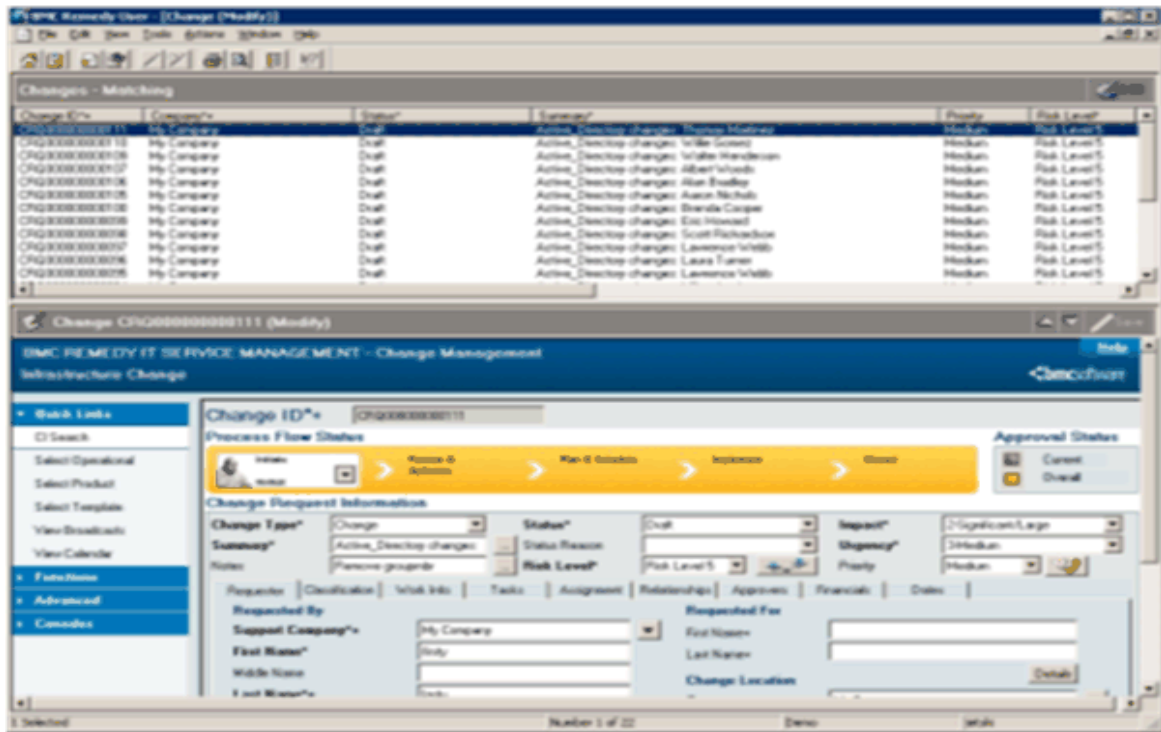
Figure 1—Change request

Configuring BMC Remedy AR System for IdentityIQ Integration

This section provides the required information for configuring IdentityIQ to integrate with BMC Remedy Action Request System (AR System). This integration enables IdentityIQ to create Change Management tickets for requested revocations, track ticket numbers in association with revocation tasks, and update IdentityIQ with the status of current Change Management tickets.

The following steps should be performed to modify the default Remedy integration configuration for a specific BMC Remedy application instance.

1. Confirm the default Remedy Change Management Application Web Services exist. This is done by launching the BMC Remedy Administrator, expanding the appropriate server object and clicking on the "Web Services" object.

2. Next, obtain the environment-specific Web Service "endpoint" by performing the following steps:
    a. Double-click on the Web Service and select the WSDL tab. Copy the WSDL handler URL into your buffer (For example, Ctrl-C)

b. With a web browser, visit the WSDL URL for the web service by entering the URL into the browser address field and pressing return.

c. Search for **soap:address location=** to find the endpoint URL. Copy this value. It will be used to replace the endpoint URL in the default IdentityIQ Remedy IntegrationConfig object.


```
- <wsdl:port binding="s0:CHG_ChangeInterface_Create_WSSoapBinding" name="CHG_ChangeInterface_Create_WSSoap">
    <soap:address location="http://jetski:8080/arsys/services/ARService?server=jetski&webService=CHG_ChangeInterface_Create_WS"/>
  </wsdl:port>
```

d. Review the Create InputMap section of the WSDL to understand the fields available for population through the Web Service. These fields should correspond to the fields listed in the <soapenv:Body> section of the default IdentityIQ IntegrationConfig object

3. Once you are familiar with the WSDL, modify the default IdentityIQ Remedy integration using the information collected about the web service.

   a. In the <IntegrationConfig> element of the integration configuration, modify the **username** and **password** entries in the attributes map to contain the credentials required for authentication to the web service.

   b. In the <IntegrationConfig> element of the integration configuration, modify the provision entry of the Attributes map by setting the endpoint, and, if necessary, the namespace, the prefix, the responseElement, and the soapMessage attributes (the default values: IdentityIQ Remedy IntegrationConfig):

      i. Set the value for endpoint to the value located in the WSDL earlier.

      Note: The value in the IdentityIQ integration configuration must be a valid HTTP URL and have any special characters escaped. The most common change that must be made is to replace all & symbols with &

      ii. The value for namespace comes from the **targetNamespace** attribute of the **xsd:schema** element in the WSDL.

iii. The value for prefix is the prefix of the XML elements that will be contained in the SOAP response sent by the mid tier server.

iv. The value for responseElement should be the ARS form field that corresponds to the id of the form that the web service creates.

v. The value for soapMessage should be the SOAP message body that IdentityIQ will send to ARS. The exact format of this message is a function of the form that is published as described by the form's WSDL. The XML elements in the **soapenv:Body** element should be changed to match the ARS form fields for the published web service. Each required ARS form field must have an element in the SOAP message. The value can be fixed or can be a variable that will be substituted using IdentityIQ's Velocity templating.

The information in the reference section above show the variables that are provided and the example integration configuration provides examples of how they are used.

Configuring IdentityIQ for BMC Remedy Action Request System Integration

This is intended as an introduction to the configuration needed to integrate **IdentityIQ** with the BMC Remedy Action Request System. This integration enables **IdentityIQ** to interact with many of the product solutions that are built on top of the AR System Server including BMC Remedy Change Management, BMC Remedy IT Service Management Suite, and BMC Remedy Service Desk.

Prerequisites

When running BMC Remedy Service Desk Service Integration on an application server using Java 6, the JRE must be directed to use the SOAP/SAAJ implementation provided by Axis2. To do this, add the following Java options to the application server:

- `-Djavax.xml.soap.SOAPConnectionFactory=org.apache.axis2.saaj.SOAPConnectionFactoryImpl`
- `-Djavax.xml.soap.MessageFactory=org.apache.axis2.saaj.MessageFactoryImpl`
- `-Djavax.xml.soap.SOAPFactory=org.apache.axis2.saaj.SOAPFactoryImpl`

Without these options, a ClassCastException error or Signature creation failed error will occur when the Remedy integration is used.

BMC Remedy Action Request System Integration

SailPoint provides a default Remedy integration configuration. This configuration implements the integration between IdentityIQ and the Remedy Change Management Application to fulfill creation of tickets based on IdentityIQ access certification remediation events.

The default configuration is located in `iiqHome/WEB-INF/config/remedy-Integration.xml` directory, where *iiqHome* is the location where IdentityIQ was installed.

For BMC Remedy AR System versions 7.5.00 patch 001, 8.0 and 8.1, an additional sample xml file is located in `iiqHome/WEB-INF/config/remedyV8-integration.xml` directory.

This section explains the various entries that are specific for this integration. For more information of the entries in the `IntegrationConfig` file, see Appendix A: Common Identity Management Integration Configuration.

The integration configuration must include the following entries:
- **endpoint**: URL to the web service
- **namespace**: namespace of the XML returned by the web service
- **prefix**: prefix associated with the namespace

The integration configuration includes the following entries if the web service side of the integration is configured for authentication using the SOAP authentication specifications:
- username
- password
- authentication
- locale
- timeZone
- statusMap

The integration configuration includes the following entries if the http authentication is configured:
- basicAuthType: if http authentication is configured the value of basicAuthType is true.
- httpUserName
- httpUserPass

User must modify remedy integration configuration file with the following entries to create incident in BMC Remedy Action Request System:
- endpoint
- responseElement key
- SOAP envelop and body details
- status mapping

The web services and authentication entries are consumed by configuration entries for each web service. They can be positioned either within the configuration entries themselves or as children of the Attributes element. Entries that are children of the Attributes element can be thought of as global values, while entries within the configuration entities can be thought of as local.

For example, if both entries share the same authentication credentials, those credentials might be placed in the Attributes element as peers of the configuration entries and the integration code searches the parent entry for the credentials if they are not found in the configuration entries. Conversely, if the configuration entries have different endpoints (are handled by separate web services), each configuration entry specifies the endpoint of the web service to call and any value outside of the configuration entry is ignored.

There are two supported configuration entries for integration with Remedy. These entries are children of the integration Attributes element:
- getRequestStatus
- provision

The values of each are Map elements containing key/value pairings of the configuration data. They contain the specific data needed by the getRequestStatus()and provision() methods of the IdentityIQ integration executor and correspond to Remedy Web Service methods.

The **getRequestStatus** and **provision** entries contain the following entries:

- **soapMessage** (required): full XML template of the entire SOAP envelope that is sent to the web service. The integration code first runs this template through Apache's Velocity template engine to provide the data needed by the web service.
- **responseElement** (required): name of the element containing the results of the web service call (for example, the element containing the ticket number opened by the web service in response to the call from IdentityIQ).
- **statusMap** (optional, see "Sample getRequestStatus entry" on page 37 for an example)
- **username** (optional)
- **password** (optional)
- **authentication** (optional)
- **locale** (optional)
- **timeZone** (optional)
- **endpoint** (optional)
- **namespace** (optional)
- **prefix** (optional)

Before a template is sent to the web service, it is processed by the **Velocity template engine**. The integration code provides different data objects to Velocity for evaluation based on the integration method.

The **provision** call passes the following objects to Velocity:

- **config**: the integration configuration for provision, represented as a Map
- **provisioningPlan**: the data model of the provision request

The **getRequestStatus** call passes the following objects to Velocity:

- **config**: the integration configuration for getRequestStatus, represented as a Map
- **requestID**: the string ID of the request whose status is being queried

Both calls have access to a timestamp variable containing a current Date object and a dateFormatter object. The dateFormatter is built using an optional dateFormat attribute from the **config** object. If the dateFormat attribute does not exist, the formatter defaults to the pattern EEE, d MMM yyyy HH:mm:ss z.

Sample getRequestStatus entry

> **Note:** The entries contained in the Map are the only required entries. Any authentication information required by this integration is inherited from the parent Attributes element.

```
<entry key="getRequestStatus">
  <value>
    <Map>
        <entry key="responseElement" value="Status"/>
        <entry key="soapMessage">
            <!-- XML template - DO NOT add line breaks before the CDATA! -->
            <value><String><![CDATA[<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
                xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
#if ($config.username)
<soapenv:Header>
<ns1:AuthenticationInfo xmlns:ns1="urn:AuthenticationInfo">
                <ns1:userName>$config.username</ns1:userName>
                <ns1:password>$config.password</ns1:password>
        #if ($config.authentication)
```

```
                       <ns1:authorization>$config.authentication</ns1:password>
#end
#if ($config.locale)
                   <ns1:locale>$config.locale</ns1:password>
#end
#if ($config.timeZone)
                   <ns1:timeZone>$config.timeZone</ns1:password>
#end
</ns1:AuthenticationInfo>
</soapenv:Header>
#end
<soapenv:Body>
       <iiq:Get xmlns:iiq="urn:GetAgreementWebService">
            <iiq:Issue_ID>$requestID</iiq:Issue_ID>
        </iiq:Get>
     </soapenv:Body>
</soapenv:Envelope>
]]>
                    </value>
                </entry>
           </Map>
         </value>
</entry>
```

Sample provision entry

> **Note:** **This Map contains its own web services information. Any authentication information required**
> **by this integration is inherited from the parent Attributes element.**

```
<entry key="provision">
      <value>
           <Map>
                    <entry key="endpoint"
value="http://my.server.com:8080/path/to/WS"/>
                    <entry key="namespace" value="urn:openTicketWebService"/>
                    <entry key="prefix" value="xyz"/>
                    <entry key="responseElement" value="Issue_ID"/>
                    <entry key="soapMessage">
                        <!-- XML template - DO NOT add line breaks before the CDATA!
-->
                        <value><String><![CDATA[<?xml version="1.0"
encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
                   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
#if ($config.username)
<soapenv:Header>
<ns1:AuthenticationInfo xmlns:ns1="urn:AuthenticationInfo">
                   <ns1:userName>$config.username</ns1:userName>
                   <ns1:password>$config.password</ns1:password>
#if ($config.authentication)
                   <ns1:authorization>$config.authentication</ns1:password>
#end
#if ($config.locale)
                   <ns1:locale>$config.locale</ns1:password>
#end
#if ($config.timeZone)
                   <ns1:timeZone>$config.timeZone</ns1:password>
```

```
#end
</ns1:AuthenticationInfo>
</soapenv:Header>
#end
<soapenv:Body>
    <iiq:Get xmlns:iiq="urn:openTicketWebService">
        <iiq:Submitter>
        #foreach ($req in $provisionPlan.requesters)
            $req.name
        #end
        </iiq:Submitter>
        </iiq:SubmitDate>$timestamp</iiq:SubmitDate>
        <iiq:Summary>
            Remediation request from IIQ
        </iiq:Summary>
        <iiq:Description>
            Remove Active Directory for $provisionPlan.identity.fullname
        </iiq:Description>
        <iiq:Issue_ID>$requestID</iiq:Issue_ID>
    </iiq:Get>
    </soapenv:Body>
</soapenv:Envelope>
]]>
        </value>
      </entry>
    </Map>
  </value>
</entry>
```

Sample statusMap entry

The **noMappingFromWS** entries are placeholders as there are no results from the web service corresponding to those IdentityIQ result codes.

```
<entry key="statusMap">
    <value>
        <Map>
            <entry key="Closed" value="committed" />
            <entry key="Rejected" value="failure" />
            <entry key="Draft" value="inProcess" />
            <entry key="Pending" value="inProcess" />
            <entry key="noMappingFromWS" value="retry" />
            <entry key="noMappingFromWS" value="warning" />
        </Map>
    </value>
</entry>
```

Creating multiple tickets in Remedy System

To create multiple tickets in Remedy System via IdentityIQ, add the following attributes in
`remedy-integration.xml` file:

- **multipleTicket**: If multipleTicket attribute is defined, then the value can be one of the following:

  - True: A separate Remedy ticket will be created for each line item from the IdentityIQ access request.

  - False: Single Remedy ticket will be created against all line items from the IdentityIQ access request.
    Default value: true

The format of the entry is as follows:

```
<entry key='multipleTicket' value='true'/>
```

- **groupTicketBy**: If groupTicketBy attribute is defined, then value can be one of the following:

    - none: If the attribute is not defined or if attribute value is other than Application, then IdentityIQ sets this attribute to none.

    - Application: If the attribute value is Application and `multipleTicket=true`, then IdentityIQ access request lines the items from the same application will be moved to a single ticket.
    The format of the entry is as follows:

```
<entry key='groupTicketBy' value='none'/>
```

Default value: none

Sample scenario

The sample integration scenario is built around a sample system. In the sample scenario SailPoint (IIQ) will be issuing a change request to BMC Remedy Change Management (RCM) based on the results of a scheduled user entitlement and access review. As a result of remediation actions in this account recertification process, IdentityIQ will open change requests to control the flow of the manual remediation process.

Scenario

1. The ComplianceManager1 schedules an access review for a business critical application:
    a. The certification is scheduled and assigned to ApplicationOwner1.
    b. ApplicationOwner1 receives an email with a link to the Online certification process as scheduled. The link is followed to the open certification.
    c. ApplicationOwner1 decides that GroupA on system LDAP should be removed.
    d. ApplicationOwner1 decides that RoleA on system RDBMS should be removed.
    e. ApplicationOwner1 completes the certification and signs off the process.

2. IdentityIQ evaluates the provisioning plan to enact the remediation requests from the certification:
    a. IdentityIQ policy describes the integration execution path for LDAP as being via an automated pro-visioning system.
    b. IdentityIQ policy describes the integration execution path for RDBMS as being via an automated RCM integration.

3. IdentityIQ creates a service request in RCM:
    a. IdentityIQ uses the **provision** interface to open a service request within Remedy, passing in details of the changes required to the RDBMS system.
    b. RCM responds with the service request number.
    c. IdentityIQ stores the service request number for later audit and review.

Chapter 7: SailPoint ServiceNow Service Integration Module

The following topics are discussed in this chapter:

Overview

The integration between SailPoint and ServiceNow enables customers to create service requests, incidents, and change requests in ServiceNow for the configured operations (for example, creating account, removing/deleting access and other operations) for the configured application. The seamless integration of SailPoint and ServiceNow Service Integration Module eliminates the need to build and maintain a custom integration, and reduces time-to-deployment.

Supported features

SailPoint ServiceNow Service Integration Module supports the following features:
- Creating ticket for all provisioning operations
- Syncing ticket status between the two systems
- Retry Mechanism for Create Ticket request failure

Supported platforms

SailPoint ServiceNow Service Integration Module supports the following ServiceNow versions:
- Geneva
- Fuji
- Eureka

Pre-requisites

Ensure that the pre-requisites for Service Request, Incident and Change Request specified in this section are performed.

Service Request

- ServiceNow Instance must be up and running.

  The ServiceNow Service Integration Module Administrator must be assigned the following role:

  - (*For Fuji or Later*) `x_sap_iiq_sim.admin`

  - (*For Eureka*) `iiq_servicenow_sim_role`

- Apply the ServiceNow Service Integration Module update set:
  - a. Copy the relevant update set from the following directory to a temporary directory:

    ```
    identityiq-releaseVersion.zip\integration\servicenow\iiqIntegration-Servi
    ceNow.zip\ServiceIntegrationModuleUpdateSet\IdentityIQServiceNowServiceIn
    tegrationModule.v2.x_serviceNowInstanceVersion.xml
    ```

  In the above directory:

  - *releaseVersion* is the version of the current IdentityIQ release

  - *serviceNowInstanceVersion* is **eureka** and empty for **Fuji or Later**

ServiceNow version	Update Sets
`Fuji or Later`	`IdentityIQServiceNowServiceIntegrationModule.v2.0.xml`
`Eureka`	`IdentityIQServiceNowServiceIntegrationModule.v2.0_eureka.xml`

  - b. Import relevant update set in ServiceNow instance. For more information and guidelines on usage of the update set, refer to the following wiki link:
    http://wiki.servicenow.com/
  - c. In case of **Fuji or Later**, import the `ServiceRequestGenerator.xml` file. Navigate to Scripted Web Services under System Web Services, right click on one of the Column header and click on **Import xml**.

- ServiceNow Integration provides the following files located in *iiqHome*/WEB-INF/config/ directory:

  - (*For Fuji or Later*) `ServiceNowServiceIntegrationModule.xml`

  - (*For Eureka*) `ServiceNowServiceIntegrationModuleEureka.xml`

Incident and Change Request

- ServiceNow Instance must be up and running.

  The ServiceNow Service Integration Module Administrator must be assigned the following role:

  - (*For Fuji or Later*) `x_sapo_iiq_sim.admin`

  - (*For Eureka*) `iiq_servicenow_sim_role`

- Apply the ServiceNow Service Integration Module update set:
  - a. Copy the relevant update set from the following directory to a temporary directory:

```
identityiq-releaseVersion.zip\integration\servicenow\iiqIntegration-Servi
ceNow.zip\ServiceIntegrationModuleUpdateSet\IdentityIQServiceNowServiceIn
tegrationModuleForIncidentAndChange.v1.x_serviceNowInstanceVersion.xml
```

In the above directory:

- *releaseVersion* is the version of the current IdentityIQ release

- *serviceNowInstanceVersion* is **eureka** and empty for **Fuji or Later**

ServiceNow version	Update Sets
Fuji or Later	IdentityIQServiceNowServiceIntegrationModuleForIncidentAndChange.v1.2.xml
Eureka	IdentityIQServiceNowServiceIntegrationModuleForIncidentAndChange.v1.6_eureka.xml

 b. Import relevant update set in ServiceNow instance. For more information and guidelines on usage of the update set, refer to the following wiki link:
http://wiki.servicenow.com/

- Ensure that `ServiceNowIntegrationExecutor` is being called: ServiceNowIntegrationExecutor class is responsible for creating and sending SOAP requests to ServiceNow. You can add a simple `System.out.println` statement in `ServiceNowServiceIntegration` rule to ensure that this is being called when a provisioning request is submitted for this integration.

- ServiceNow Integration provides the following files located in `iiqHome`/WEB-INF/config/ directory:

   - (*For Fuji or Later*) `ServiceNowSIMForIncidentAndChange.xml`

   - (*For Eureka*) `ServiceNowSIMForIncidentAndChangeEureka.xml`

When integrating with the following service operations, open and modify the respective sections in the above files and import in IdentityIQ:

   - for Incident: #INCIDENT

   - for Change Request: #CHANGE REQUEST

For more information, see "Configuring IdentityIQ to integrate with ServiceNow" on page 46.

Note: **If the field is a reference field and the reference value is other than those available, then ServiceNow implicitly creates those reference records. If you do not want to allow creation of reference records, then set** `Choice action` **to** `ignore` **in the transform map for the reference field.**

Basic configuration

The integrated solution speeds the detection and remediation of identity management issues that increase the risk of compliance violations or security breaches, such as orphaned accounts, policy violations, and inappropriate access privileges. Organizations can take advantage of a centralized approach spanning thousands of users and hundreds of resources to strengthen IT controls and provide proof of compliance to auditors and executive management. The seamless integration of SailPoint and ServiceNow eliminates the need to build and maintain a custom integration, and speeds time-to-deployment.

For any IT resources managed by ServiceNow Service Desk, IdentityIQ automatically creates a trouble ticket within ServiceNow Service Desk, passing along all relevant identity data and reviewer comments to populate the ticket.

To ensure revocation requests get delivered and implemented, IdentityIQ manages all remediation and revocation requests within a guaranteed delivery model.

To determine the status of user accounts, IdentityIQ performs closed-loop audits on remediation requests and compares the actual state of user privileges with the original change request. If the request is still open, an alert will be sent to the reviewer for prompt action and closure.

The integration itself has been designed to be quick to install and easy to use. It makes use of Web Services for communications between the SailPoint server and the ServiceNow. On the backside of a user recertification, policy remediation action or access request action, the IdentityIQ server will direct provisioning and service desk requests to the configured implementers. Based on the `IntegrationConfig` configured for each target application, service desk request are issues to a given remediation/implementation point. Once the `IntegrationConfig` file for ServiceNow has been loaded into the IdentityIQ server, all change/remediation actions result in the creation of new service desk request as shown in Figure 1—Basic configuration.

The SailPoint ServiceNow Service Integration Module generates tickets for provisioning requests. These tickets generate service requests on `sc_request` and `sc_req_item` table, incidents on `incident` table, or change requests on the `change_request` table. The module fetches the status of ticket by using the direct web services of target tables that is, `sc_req_item, incident` or `change_request` and updates the SailPoint IdentityIQ database with the status.
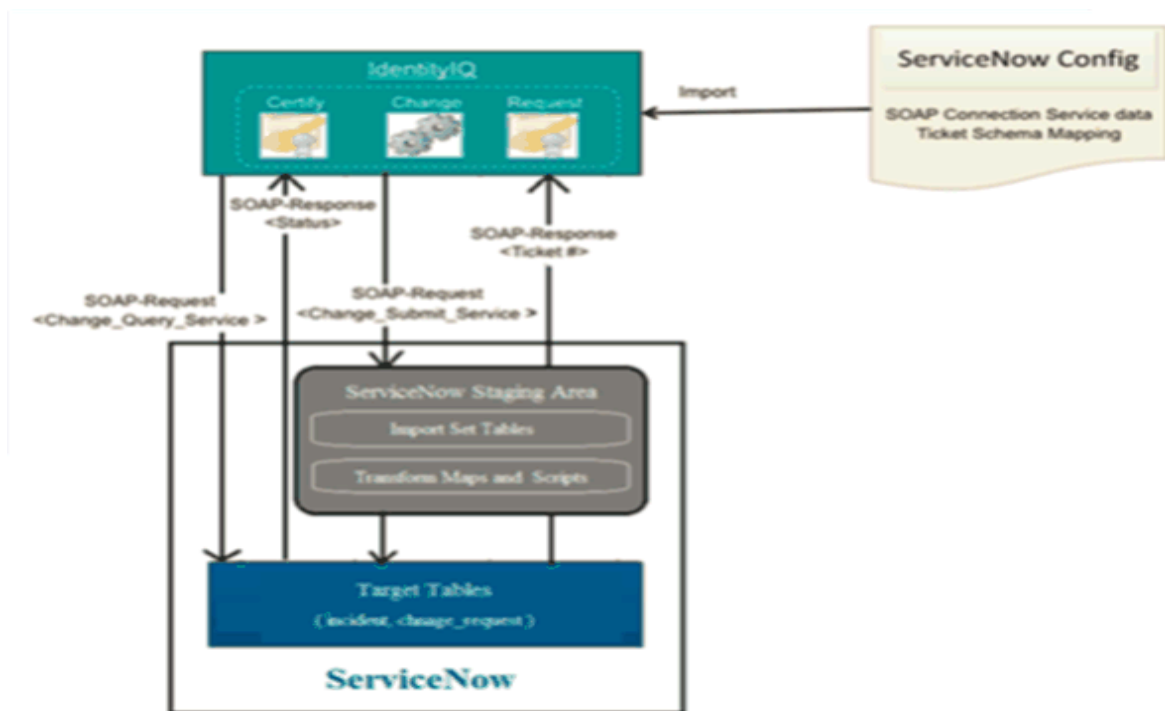
Figure 1—Basic configuration

At the completion of the change control cycle within IdentityIQ, an "Open Ticket" request is made over the appropriate SOAP channel to the ServiceNow web service. From here, tickets are opened and the new ticket number is returned to IdentityIQ. The schema for the service request is defined in the `IntegrationConfig` and allows for the flexibility to transfer complete details on the service desk request.

Basic flow of Service Request

IdentityIQ creates ticket by requesting a Catalog item on ServiceNow. Each application on IdentityIQ has a Catalog item defined on ServiceNow. IdentityIQ calls ServiceNow's Web Service requesting the Catalog Item. The Web service creates a Cart and adds the Catalog item to the cart. The Catalog item has Catalog Variables. The information is taken from the IdentityIQ's request and passed on to these Catalog Variables. The Catalog Item has a Workflow attached to it. After adding Catalog Item to the cart, Cart is submitted. Submission of cart triggers the Workflow. The workflow creates a task by passing the information from Catalog Variables to Service Request. The Requested Item ticket number is returned as the response which is later used to check the status.

Depending on the workflow configuration, the task is assigned to the user (group or individual), who then performs the action which results in change in the State of the Requested Item.

Basic configuration of Service Request

- Create Catalog items on ServiceNow for each application on IdentityIQ that user wants to manage using ServiceNow Service Integration Module. The name must contain "IdentityIQ" to filter and display in SailPoint Service Integration Module under Catalog Items.

  For more information on the procedure for creating the catalog item, see the following link:

  http://wiki.servicenow.com/index.php?title=Defining_Catalog_Items

  Note: Refer to the default Catalog Items provided in the update set.

- Catalog Variables must be created for each element defined in SOAP message in ServiceNow Integration Configuration file. The name of the Catalog variable must be same as the respective SOAP element in the SOAP message.

  For more information on the procedure for creating the catalog variable, see the following link:

  http://wiki.servicenow.com/index.php?title=Using_Service_Catalog_Variables

- The **opened_by** and **requested_for** fields receive values using the default Rule in ServiceNow Configuration file. Modify the rule as per requirement.

  For example, see **requested_for** catalog variable in the default Catalog item (IdentityIQ Access Request).

  Note: If the user is not present on ServiceNow, then 'opened_by' and 'requested_for' fields will display default ServiceNow Administrator.

- Provide the mapping between Application and Catalog item in the **catalogItem** field in ServiceNow Integration Configuration file.

  For example, `<entry key="Procurement_System" value="Procurement_System Item"/>`

  Note: The Catalog item name is case sensitive.

- Verify the default mapping between ServiceNow's Request Item (RITM) Status field and IdentityIQ status in ServiceNow Integration Configuration file. This is used to update the status of IdentityIQ line item.

- Create and publish a workflow and attach it to the respective Catalog Item to handle the Requested Item (RITM). The name must contain "IdentityIQ" to filter and display in SailPoint Service Integration Module under Workflow versions.

  The workflow must be able to create a Catalog Task.

  For more information on procedure for creating the workflow, see the following link:

  http://wiki.servicenow.com/index.php?title=Creating_a_Workflow

> Note: Refer the default workflow given in the update set. Login with the Admin user or user with 'workflow_admin' to create or update workflow.

- The default workflow, in the **Run Script** activity has a scripting logic to get values from Catalog Variables and assign to the fields of Service Request and Requested Item. Use that or modify accordingly. Configure the Workflow as per requirement.

- The **State** of Requested Item (RITM) changes when the **State** field of the Task changes due to the Workflow defined on the Catalog Item. The **Request State** on the Service Request changes due to the default 'Service Catalog Request' workflow.

- If some records or columns are not displayed with Minimum permission user, add the specific ACLs to view the records.

  For more information on procedure for creating the ACL, see the following link:

  http://wiki.servicenow.com/index.php?title=Using_Access_Control_Rules#Creating_ACL_Rules

Configuring IdentityIQ to integrate with ServiceNow

This section provides the required information for configuring IdentityIQ to integrate with ServiceNow.

This is intended as an introduction to the configuration needed to integrate IdentityIQ with ServiceNow. It outlines some examples that must be used as a reference point for implementation. Some changes may be required to meet specific use case and expertise around both systems are a must for the successful implementations.

SailPoint provides a default ServiceNow configuration. This configuration implements the integration between IdentityIQ and the ServiceNow to fulfill creation of tickets based on IdentityIQ access certification remediation events.

The default configuration is located in the following directory, where *iiqHome* is the location where IdentityIQ was installed:

- Service Request

  - (*For Fuji or Later*)
    `iiqHome/WEB-INF/config/ServiceNowServiceIntegrationModule.xml`

  - (*For Eureka*)
    `iiqHome/WEB-INF/config/ServiceNowServiceIntegrationModuleEureka.xml`

- Incident and Change Request

  - (*For Fuji or Later*) `iiqHome/WEB-INF/ServiceNowSIMForIncidentAndChange.xml`

  - (*For Eureka*) `iiqHome/WEB-INF/ServiceNowSIMForIncidentAndChangeEureka.xml`

This section explains the various entries that are specific for this integration. For more information of the entries in the `IntegrationConfig` file, see Appendix: A: Common Identity Management Integration Configuration.

The integration configuration must include the following entries:

- **endpoint**: URL to the web service

Endpoint task	Service operation	Eureka	Fuji or Later
ServiceNow endpoint to create a ticket	Service Request	**https://demo.service-now.com /ScRequestGenerator.do?SOAP**	**https://demo.service-now.com /ScRequestGenerator.do?SOAP**
	Incident	**https://demo.service-now.com /u_sailpointincident.do?SOAP**	**https://demo.service-now.com /x_sapo_iiq_sim_incident.do? SOAP**
	Change Request	**https://demo.service-now.com /u_sailpointchangerequest.do? SOAP**	**https://demo.service-now.com /x_sapo_iiq_sim_change_requ est.do?SOAP**
ServiceNow endpoint to get ticket status	Service Request	**https://demo.service-now.com/sc_req_item.do?SOAP**	
	Incident	**https://demo.service-now.com/incident.do?SOAP**	
	Change Request	**https://demo.service-now.com/change_request.do?SOAP**	

- **namespace**: namespace of the XML returned by the web service
- **prefix**: prefix associated with the namespace

The integration configuration includes the following entries if the web service side of the integration is configured for authentication using the SOAP authentication specifications:

- username (Can be blank if **Require basic authorization for incoming SOAP requests** is not selected on ServiceNow side)
- password (Can be blank if **Require basic authorization for incoming SOAP requests** is not selected on ServiceNow side)
- authentication
- locale
- timeZone
- statusMap

The integration configuration also includes the following properties if WS-Security is enabled on service-now side:

- authType
- keystorePath
- keystorePass
- keystoreType
- alias
- keyPass
- catalogItem (For Service Request)

For more information on enabling the WS-Security on ServiceNow side, see "Configuration required on ServiceNow side for WS-Security" on page 51.

The web services and authentication entries are consumed by configuration entries for each web service. They can be positioned either within the configuration entries themselves or as children of the Attributes element. Entries that are children of the Attributes element can be thought of as global values, while entries within the configuration entities can be thought of as local.

For example, if both entries share the same authentication credentials, those credentials might be placed in the Attributes element as peers of the configuration entries and the integration code searches the parent entry for

the credentials if they are not found in the configuration entries. Conversely, if the configuration entries have different endpoints (are handled by separate web services), each configuration entry specifies the endpoint of the web service to call and any value outside of the configuration entry is ignored.

There are two supported configuration entries for integration with ServiceNow. These entries are children of the integration Attributes element:

- getRequestStatus
- provision

The values of each are Map elements containing key/value pairings of the configuration data. They contain the specific data needed by the **getRequestStatus()** and **provision()** methods of the IdentityIQ integration executor and correspond to ServiceNow Web Service methods.

The **getRequestStatus** and **provision** entries contain the following entries:

Entries	Description
soapMessage*	Full XML template of the entire SOAP envelope that is sent to the web service. The integration code first runs this template through Apache's Velocity template engine to provide the data needed by the web service.
responseElement*	name of the element containing the results of the web service call (for example, the element containing the ticket number opened by the web service in response to the call from IdentityIQ).
statusMap	For example, see "IntegrationConfig XML files for Service Request, Incident and Change Request" on page 49.
username	Can be blank if **Require basic authorization for incoming SOAP requests** is not selected on ServiceNow side.
password	Can be blank if **Require basic authorization for incoming SOAP requests** is not selected on ServiceNow side.
authentication	
locale	(*Optional*)
timeZone	(*Optional*)
endpoint	(*Optional*)
namespace	(*Optional*)
prefix	(*Optional*)
authType	(*Optional*) Use **"WS-Security"** if WS Security is enabled on service-now side. Otherwise leave it blank.
keystorePath	(*Optional*) Full path of keystore.
keystorePass	(*Optional*) Password of keystore.
keystoreType	(*Optional*) Type of keystore. For example, jks
alias	(*Optional*) The alias of certificate in keystore.
keyPass	(*Optional*) The password of alias.
catalogItem	(*For Service Request only*) Map of Catalog items on ServiceNow defined for IdentityIQ applications.

Before a template is sent to the web service, it is processed by the **Velocity template engine**. The integration code provides different data objects to Velocity for evaluation based on the integration method.

The **provision** call passes the following objects to Velocity:

- **config**: the integration configuration for provision, represented as a Map
- **provisioningPlan**: the data model of the provision request

The **getRequestStatus** call passes the following objects to Velocity:

- **config**: the integration configuration for getRequestStatus, represented as a Map
- **requestID**: the string ID of the request whose status is being queried

Both calls have access to a timestamp variable containing a current Date object and a dateFormatter object. The dateFormatter is built using an optional **dateFormat** attribute from the **config** object. If the dateFormat attribute does not exist, the formatter defaults to the pattern EEE, d MMM yyyy HH:mm:ss z.

IntegrationConfig XML files for Service Request, Incident and Change Request

The entries contained in the Map are the only required entries. Any authentication information required by this integration is inherited from the parent Attributes element.

For more information and examples of the sample files, see the following sample files:

- Service Request

    - (*For Fuji or Later*) ServiceNowServiceIntegrationModule.xml

    - (*For Eureka*) ServiceNowServiceIntegrationModuleEureka.xml

- Incident and Change Request

    - (*For Fuji or Later*) ServiceNowSIMForIncidentAndChange.xml

    - (*For Eureka*) ServiceNowSIMForIncidentAndChangeEureka.xml

The IntegrationConfig.xml file provides configuration for the following ServiceNow service operations:

- Service Request
- Incident
- Change Request

    Note: **The IdentityIQ integration for 'Service Request' service operation uses Requested Item (RITM) based approach.**

If any changes required in the mapping, change the default value/key values in **statusMap** and **statusMapCloserCode** as mentioned in the following tables:

- statusMap for Service Request:

Entry key (ServiceNow)	Value (IdentityIQ)
-5	inProcess
1	inProcess
2	inProcess
4	failure

Entry key (ServiceNow)	Value (IdentityIQ)
7	failure
3	committed

- statusMap and statusMapCloserCode for Incident

Entry key (ServiceNow)	Values (IdentityIQ)
statusMap	
1	inProcess
2	inProcess
3	inProcess
4	inProcess
5	inProcess
6	committed
7	committed
statusMapCloserCode	
Solved (Work Around)	committed
Solved (Permanently)	committed
Solved Remotely (Work Around)	committed
Solved Remotely (Permanently)	committed
Closed/Resolved by Caller	committed
Not Solved (Not Reproducible)	failure
Not Solved (Too Costly)	failure

- statusMap for Change Request

Entry key (ServiceNow)	Value (IdentityIQ)
-1	inProcess
1	inProcess
2	inProcess
3	committed
4	failure
7	failure

Configuration procedure

The following steps should be performed to modify the default ServiceNow integration configuration for a specific ServiceNow instance.

1. Obtain the environment-specific Web Service "endpoint", for example, https://demo.service-now.com/incident.do?SOAP
   Either a web service can be created a web service pointing to system table can be used, for example, https://demo.service-now.com/incident.do?SOAP

2. Once you are familiar with the WSDL, modify the default IdentityIQ ServiceNow configuration using the information collected about the web service.
   a. In the <IntegrationConfig> element of the integration configuration, modify the **username** and **password** entries in the attributes map to contain the credentials required for authentication to the web service.
   b. If you have enabled WS-Security on ServiceNow side, modify entries for **authType**, **keystorePath**, **keystorePass**, **keystoreType**, **alias**, **keyPass** to contain keystore related details.
   c. In the <IntegrationConfig> element of the integration configuration, modify the provision entry of the Attributes map by setting the endpoint, and, if necessary, the namespace, the prefix, the responseElement, and the soapMessage attributes (the default values: IdentityIQ ServiceNow IntegrationConfig):

      i. Set the value for endpoint to the value located in the WSDL earlier.

      Note: **The value in the IdentityIQ integration configuration must be a valid HTTP URL and have any special characters escaped. The most common change that must be made is to replace all & symbols with &**

      ii. The value for namespace comes from the **targetNamespace** attribute of the **xsd:schema** element in the WSDL.

      iii. The value for prefix is the prefix of the XML elements that will be contained in the SOAP response.

      iv. The value for responseElement should be the ServiceNow form field that corresponds to the id of the form that the web service creates.

      v. The value for soapMessage should be the SOAP message body that IdentityIQ will send to ServiceNow. The exact format of this message is a function of the form that is published as described by the form's WSDL. The XML elements in the **soapenv:Body** element should be changed to match the ServiceNow form fields for the published web service. Each required ServiceNow form field must have an element in the SOAP message. The value can be fixed or can be a variable that will be substituted using IdentityIQ's Velocity templating

   The information in the reference section above show the variables that are provided and the example integration configuration provides examples of how they are used.

Configuration required on ServiceNow side for WS-Security

Perform the following steps to enable WS-Security on ServiceNow side:

1. Login to service-now instance with user having access to do system changes for example, admin.

2. Navigate to **System Definition => Certificates** and click on **New** button.

3. Enter some name for the certificate.

4. In command prompt, navigate to directory where the certificate is located.

5. Enter the following command, where **mycert.pem** is the name of the certificate:

type mycert.pem

The following similar output is displayed:

```
-----BEGIN CERTIFICATE-----
MIIDOTCCAiECBE/6tzswDQYJKoZIhvcNAQEEBQAwYTELMAkGA1UEBhMCSU4xCzAJBgNVBAgTAk1I
...
...
-----END CERTIFICATE-----
```

6. Copy all above content (including Begin and End Certificate lines) and navigate to service-now and paste the above contents into **PEM Certificate** field.

7. Save the certificate.

8. Navigate to System **Web Services =>WS Security Profiles** and click on **New**.

9. In **Type** field, select X509 and in **X509 Certificate** field, select certificate which we uploaded.

10. Select bind session checkbox.

11. In Run as user field, select name of user on behalf of whom, you want to execute web-services for example, System Administrator.

12. Click on **Submit** button.

13. Navigate to **System Web Services => Properties**.

14. Select the **Require WS-Security header verification for all incoming SOAP requests** check box and save it.

Retryable mechanism

By default ServiceNow Service Integration Module provides retry mechanism for Connection reset and for unknown host problems occurred from network issues.

However you can configure **retryableErrors** list in integration configuration (`IntegrationConfig`) file to add new exception strings to the attributes map in integration configuration file.

The **retryableErrors** entry is a list of strings through which the integration searches when it receives a message from the Service Integration Module. Only **SOAPException** strings are considered for retry that is, the exceptions raised from SOAP web service. If one of the strings in the entry exists in the error, the integration attempts to retry the request. When the configured error string is not a part of the error message returned from the Service Integration Module, then IdentityIQ will not attempt a retry.

For example,

```
<entry key="retryableErrors">

    <value>

      <List>

        <String>Connection reset</String>

      </List>

    </value>

</entry>
```

> **Note:** Error messages containing very specific information about date/time, sequence ID and so on must be avoided. Error codes or error message substrings would be good candidates for inclusion. Only exceptions raised from soap web service are considered for retry.

Upgrade

Perform the following procedure for upgrading IdentityIQ ServiceNow Service Integration Module for Service Request from version 7.0 to 7.0 Patch 2:

1. IdentityIQ ServiceNow Service Integration Module update set for Service Request must be applied on ServiceNow instance.
    For more information, see "Pre-requisites" on page 42.

2. Ensure that the ServiceNow Service Integration Module user must be assigned the appropriate Service Integration Module Administrator role. For more information, see "Pre-requisites" on page 42.

3. Delete the earlier IntegrationConfig object for 'ServiceNow Integration' form IdentityIQ debug pages.

4. On the IdentityIQ side the ServiceNow Service Integration Module Integration Configuration file for Service Request must be re-imported after performing the corresponding changes.
    For more information about the Integration Configuration files, see "Configuring IdentityIQ to integrate with ServiceNow" on page 46.

Sample scenario

The sample integration scenario is built around a sample system as shown in the following figure:
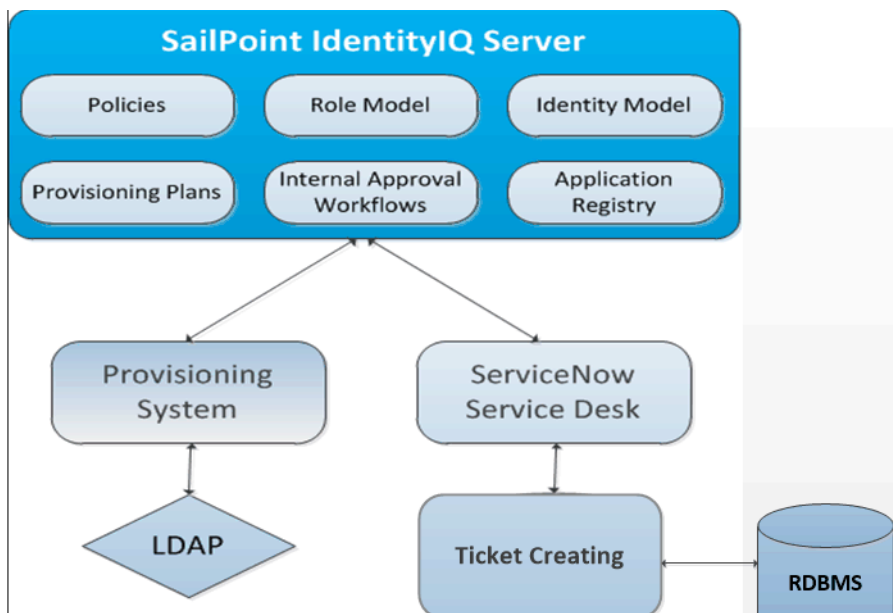
In the sample scenario SailPoint IdentityIQ will be issuing change request to ServiceNow based on the results of a scheduled user entitlement and access review. As a result of managing user access process, IdentityIQ will open ServiceNow tickets to control the flow of the manual fulfillment process.

Scenario

1. The LifeCycleManager request for the access of an employee (Identity) on business critical applications.

2. IdentityIQ evaluates the provisioning plan to enact the access requests required for the user:
   a. IdentityIQ policy describes the integration execution path for LDAP as being via an automated provisioning system.
   b. IdentityIQ policy describes the integration execution path for RDBMS as being via an automated ServiceNow Service Integration Module.

3. IdentityIQ creates a ticket in ServiceNow:
   a. IdentityIQ uses the **provision** interface to open a ticket within ServiceNow, passing in details of the changes required to the RDBMS system.
   b. ServiceNow responds with the ticket number.
   c. IdentityIQ stores the ticket number under the Access Request for later audit and review.

4. IdentityIQ synchronizes ticket status:
   a. The ticket is assigned to the appropriate ServiceDesk user/ group in the ServiceNow.
   b. The status of ticket in ServiceNow is updated by ServiceDesk user/ group.
   c. IdentityIQ synchronizes the status of ticket from ServiceNow and update the access request items status.

Troubleshooting

This section provides the resolutions for the following errors that may be encountered while setting up and configuring ServiceNow Service integration Module.

1 - 'Authorization Required' error messages

The following type of error messages appear when the authorization data is not sent to ServiceNow:

```
Caused by: org.apache.axis2.AxisFault: Transport error: 401 Error:
Authorization Required
```

Resolution: Verify the procedure to configure appropriate authorization mechanism. For more information on the procedure, see "Configuration procedure" on page 50.

2 - Document Type Declaration (DTD) parsing errors

The DTD parsing errors appear when the following JVM arguments are not defined for your application server:
- `-Djavax.xml.soap.SOAPConnectionFactory=org.apache.axis2.saaj.SOAPConnectionFactoryImpl`
- `-Djavax.xml.soap.MessageFactory=org.apache.axis2.saaj.MessageFactoryImpl`
- `-Djavax.xml.soap.SOAPFactory=org.apache.axis2.saaj.SOAPFactoryImpl`

Resolution: Ensure that the application is pointing to the correct java runtime (that is, 1.6) and the above mentioned JVM arguments are defined for application server.

> **Note:** Enter the following command to enable log4j tracing on this component:
> `log4j.logger.sailpoint.integration.servicenow =debug`**,** file

3 - For certificate based authentication the IdentityIQ Server and ServiceNow instance must have correct time set.

For certificate based authentication, ensure that the IdentityIQ Server and ServiceNow instance have the correct date, time, and timezone set.

4 - When the Test Connection fails error messages are displayed in IdentityIQ and log file of WebSphere

The following error message is displayed in IdentityIQ:

```
Unable to engage module: rampart
```

The following error message is displayed in the log file of WebSphere:

```
ERROR WebContainer: apache.axis2.deployment.ModuleDeployer:113 - The
rampart-1.6.1.mar module, which is not valid, caused Could not initialize
class org.apache.axis2.deployment.util.TempFileManager
```

Resolution: If the above error message is displayed in the log file of WebSphere, set the temporary directory in **Generic JVM arguments** of **Java Virtual Machine** by setting the following variable:

```
-Djava.io.tmpdir=<FullPathOfTempDir>
```

> **Note:** Ensure that the UNIX user where WebSphere is installed should be the owner of the temporary directory.

5 - Fields on ServiceNow side are not completed displayed.

The value of the fields in Incident/Change Request tickets on ServiceNow side are not completed displayed but truncated.

Resolution: If the value of the fields in the Incident / Change Request tickets are displayed as truncated, then increase the **Max length** values for the respective **Column name** of `u_sailpointincident` / `u_sailpointchangerequest` tables.

For example, u_description (Decription) to   4,000

u_short_description (Short decription) to   200

Chapter 8: SailPoint BMC Remedy Service Request Management Adapter

The following topics are discussed in this chapter:

Overview

The SailPoint BMC Remedy Service Request Management Adapter enables the use of Lifecycle Manager within Service Request Management (SRM) user interfaces.

The inclusion of the SailPoint BMC Remedy Service Request Management Adapter provides the following functionality within SRM:

- Enable, Unlock Accounts
- Request Entitlements
- Request IdentityIQ Roles
- Request Password Changes

> **Note:** **Any approvals or role request provisioning form configurations must be performed within the IdentityIQ interface.**

When IdentityIQ receives a request from SRM, approvals, forms or other business processes that take place in IdentityIQ are visible through the request system's ticket updates. By default SRM has visibility to the following IdentityIQ business processes:

- **Process Started** - Ticket opened.
- **Approvals** - Tickets are updated as approvals take place.
- **Provisioning Performed** - Tickets are updated after provisioning completes.
- **Process Complete** - Notification is sent when the process ends for any reason. The notice includes information including errors, events, and overall status.

This chapter provides the installation and configuration steps to deploy the SailPoint Adapter into an existing infrastructure running BMC Remedy Service Request Management. This chapter is intended for SRM and IdentityIQ System Administrators and assumes a high degree of technical knowledge of these systems.

Supported platforms

SailPoint Adapter supports BMC Remedy Action Request System version 7.6.04 of BMC Identity Management Suite.

Configuring BMC Remedy Service Request Management for IdentityIQ Integration

This section provides the required information for configuring IdentityIQ to integrate with BMC Remedy Service Request Management.

Install

1. Copy the `IIQ_IQ6.4.00.AZ` file to a folder on the BMC Remedy Action Request (AR) server. This file resides in the following location where `installDir` represents the directory in which you expanded the IdentityIQ product from the media with which it was delivered:

   `installDir`/`integration/BMCIRM/`IIQ_6.4.00.AZ

2. Run the following command and parameters from the directory in which the Adapter_Extractor resides on the AR server:

   AZExtraction.bat *ARserver port user password fullyqualifiedPathToAZfile mode*

   For example, **AZExtraction.bat pontoon 0 Demo sp06!#Hth c:\auser\iiq_7.6.04.AZ overwrite_all**

3. Restart the Action Request Server.

Configure

Use the following steps to configure the adapter in the AR environment.

1. Access the Application Administration Console.

2. Navigate to **Custom Configuration**=> **Identity Request Management**.

3. Click to expand **Adapter Configuration**.

4. Double-click **Adapter Configuration**.

5. Select the IdentityIQ Adapter from the first table.

6. Click the **Add Company - Site Configuration** option.

7. Fill in the parameters. See "Identity Request Management (IRM) Configuration Parameters" on page 59 for further information.

8. Click **Save Company - Site Configuration** to save the parameters.

9. Restart the Action Request server.

Identity Request Management (IRM) Configuration Parameters

There are three basic types of parameters:

- **Required connection**

  - **URL**: The URL of your IdentityIQ instance. For example, **http://pontoon:8080/iiq/**

  - **user**: Default is admin. This is the IdentityIQ proxy account on which to run services.

  - **Password**: The password for the IdentityIQ proxy account.

- **Workflow**: These are the workflows that are called during the various activities performed in IRM.

  - **accountWorkflow**: Specifies the workflow to call when performing unlock, enable and revoke all tasks. Default setting is **LCM Provisioning**.

  - **entitlementWorkflow**: Called when requests for role additions and removals are initiated. Default setting is **LCM Provisioning**.

  - **passwordWorkflow**: Called when password management requests are made through IRM. Default setting is **LCM Manage Passwords**.

  - **entitlementApprovalScheme**: Default setting is the owner.

  - **accountApprovalScheme**: Default setting is **None**.

  - **workflowTrace**: Default setting is **False**. Change value to **True** to enable IdentityIQ workflows to generate trace reports when executed.

  - **ticketingApplication**: The name of the application which is in conjunction with ticket updates

  - **doTicketingSystemNotification**: Provides added logic based on source, or other variables used to determine if tickets should be managed through a called subprocess. Default setting is **FALSE**.

- **Subprocess Workflow Variables**: The following are additional processes called when creating and updating an SRM ticket.

  - **project**: The current project, can be used by the rule when building the ticketing system provisioning plan.

  - **application:** The name of the application that represents the ticketing system.

  - **identityRequest**: Can be used by the rule to build up the ticket data, and is updated with the external ticket ID when one is returned by the call to the provisioning ticket.

  - **ticketDataGeneratorRule**: Rule that is launched to convert an IdentityRequest object and any additional workflow variables into a map of name /attributes for the SRM ticket.

  - **action**: An identifier which provides context to how the subprocess is called. The create method is typically only called once. The update call is performed after an action completes. This guides the rule on how to form the plan.

- **Launch** - The following settings specify which URL is called when the functions are executed within IRM.

  - **enableStartPageURL**: This is the page that is launched during the enable launch command. Default setting is `lcm/manageAccounts.jsf`.

  - **unlockStartPageURL**: Defaults to `lcm/manageAccounts.jsf`. This is the page that is launched during the unlock launch command.

  - **passwordStartPageURL**: Defaults to `lcm/managePasswords.jsf`. This is the page that is launched during the password launch command.

- **encodeURLParameters**: Defaults to **True**. This is used when launching the URLs, and when enabled the values on the URL are Base64 encoded.

Verify

Use the following steps to verify the installation and configuration:

1. Login to the BMC Remedy Action Request System as a user with request privileges

2. Access the Application tab.

3. Click **Service Request Management** to reveal its pop-up menu.

4. Click **Request Entry**.

5. Use the term "identity" in the search field to populate the Services window with identity management-related options.

6. Click **Request Access Right**.

If a separate window launches displaying the IdentityIQ user interface then the installation is deployed successfully. In the event of an error, verify the steps described in "Install" on page 58 and "Configure" on page 58 and see "Troubleshooting" on page 60 below.

Troubleshooting

If issues occur during installation or normal operation, SailPoint Support Engineers are available to assist. One valuable piece of information used in the troubleshooting process is a stack trace. This section provides the steps for enabling the stack trace capability for both IRM and IdentityIQ.

1 - Enable Stack Tracing for IRM

1. Locate the `log4j_pluginsvr.xml` file in the IRM installation directory. Default location is `C:\Program Files\BMC Software\ARSystem\pluginsvr\log4j_pluginsvr.xml`.

2. Define an appender that defines where the log/trace file resides.

   Note: **Failure to define where the log/trace file resides results in using the default location which stores massive amounts of data.**

   The following is an example appender:

```
<appender class="org.apache.log4j.RollingFileAppender" name="IIQPLUGINLOG">
<param name="File" value="C:/temp/iiqplugin.log"/>
<param name="MaxFileSize" value="999KB"/>

                <!-- Keep two backups -->

<param name="MaxBackupIndex" value="2"/>
        <layout class="org.apache.log4j.PatternLayout">
<param name="ConversionPattern" value="%d %-5p [%t] %C (%F:%L)- %m%n"/>
        </layout>
</appender>
```

3. Define a log level and appender for `iiq.integration`.

   The following is an example log level definition and appender:

```
<logger name="sailpoint.integration">
    <level value="trace"/>
    <appender-ref ref="IIQPLUGINLOG"/>
</logger>
```

2 - Enable Stack Tracing for IdentityIQ

Set the **workflowTrace** attribute to **TRUE** when configuring the IRM adapter parameters to enable logging. This makes any IdentityIQ provisioning actions traceable.S

Assisted Deployment Integration Modules

This section contains information on the following sections:

- "Provisioning Integration Modules"
- "Service Management Integration Modules"
- "Mobile Device Management Integration Modules"
- "IT Security Integration Modules"
- "SAP GRC Integration Module"

Assisted Deployment Integration Modules:
Provisioning Integration Modules

IdentityIQ can be configured to integrate with provisioning providers to automate access management for your implementation. Provisioning providers can be configured to communicate user and account information and automatically add or revoke access.

This section contains information on the following:

> **Note:** **A minority of SailPoint customers have deployed the Integration Module in this section. SailPoint will provide assistance during the deployment of this integration. Additional troubleshooting, diagnostic, and best practice information beyond what is contained in this document will be provided on Compass, SailPoint's Online customer portal. In some instances, SailPoint will guide the deployment team and actively participate in the design, configuration, and testing of the integration to the managed system.**
> **For more specific information, refer to the Connector and Integration Deployment Center on Compass.**

Chapter 9: SailPoint Microsoft Forefront Identity Manager Provisioning Integration Module

The following topics are discussed in this chapter:

Overview

> **Note:** **SailPoint will provide assistance during the deployment of this Integration Module. Additional troubleshooting, diagnostic, and best practice information beyond what is contained in this document will be provided in the Connector and Integration Deployment Center on Compass.**

This chapter provides a guide to the integration between Microsoft® Forefront® Identity Manager (FIM) and SailPoint IdentityIQ. This chapter is intended for FIM and IdentityIQ System Administrators and assumes a high degree of technical knowledge of these systems.

To execute request and response process, the request from IdentityIQ to FIM follows the following flow diagram:
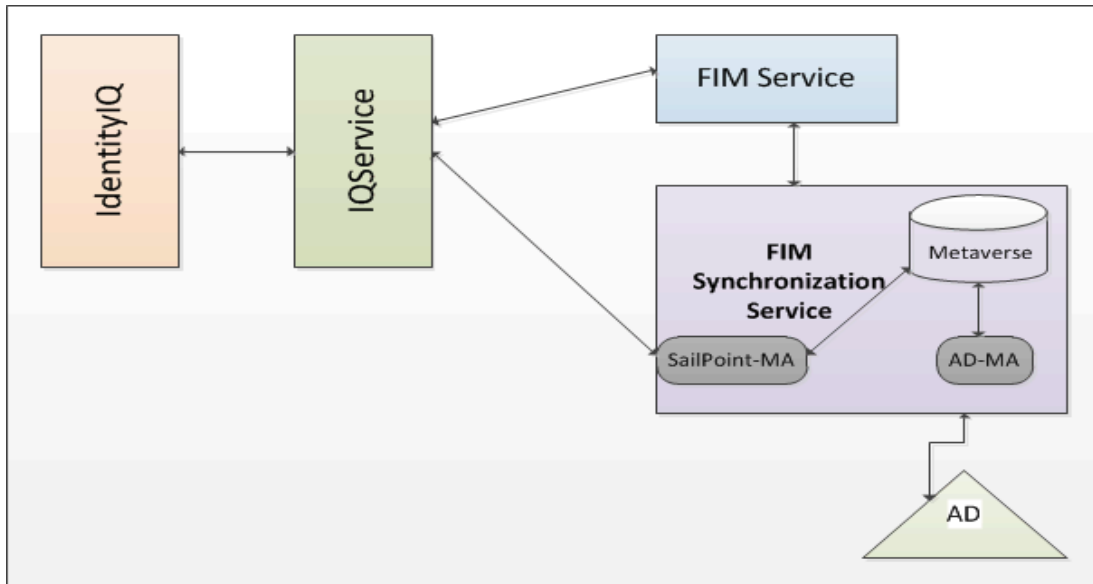
The following table displays a comparison between terminology used in IdentityIQ and similar objects in Microsoft Forefront Identity Manager:

Microsoft Forefront Identity Management	IdentityIQ
Metaverse Person	Identity
Metaverse Group	An account group for the IdentityIQ applications created to correspond to the Microsoft Forefront Identity Manger Synchronization Engine. (Main application in IdentityIQ)
Account	An account for the IdentityIQ applications created for the corresponding Management Agents of Microsoft Forefront Identity Manger. (Child Application in IdentityIQ)
Group	An account group for the IdentityIQ applications created for the corresponding Management Agents of Microsoft Forefront Identity Manger. (Child Application in IdentityIQ)

Note: **Microsoft Forefront Identity Manager Administrator must configure Metaverse Extension rule or Synchronization rule to provision data from Metaverse to SailPoint-MA connector space. In addition to this, Administrator must configure required rules to project SailPoint-MA connector space data to Metaverse and to other Management Agent connector space. For more information see, "Operation specific configuration on Microsoft Forefront Identity Manger" on page 73.**

Supported features

The Microsoft Forefront Identity Manager Provisioning Integration Module provides the ability to provision Microsoft Forefront Identity Manager Metaverse users, Target Application accounts, and entitlements from IdentityIQ.

The Microsoft Forefront Identity Manager Provisioning Integration Module supports the following functions:

- User Management

    - Create, Update, Delete

    - Enable, Disable, Unlock, Reset Password

    - Aggregation, Delta Aggregation

- Target Application Accounts Management

    - Create, Update, Delete

    - Enable, Disable, Unlock, Reset Password

    - Aggregation, Delta Aggregation

- Account - Group Management

    - Aggregation, Delta Aggregation for Groups

    - Aggregation, Delta Aggregation for target application Groups

    - Add/Remove Entitlement for Users

    - Add/Remove Entitlement for target application Accounts

Supported platforms

SailPoint Microsoft Forefront Identity Manager Provisioning Integration Module supports the following versions of Forefront Identity Manager:

- Microsoft® Forefront® Identity Manager 2010 R2
- Microsoft® Forefront® Identity Manager 2010

Configuring Microsoft Forefront Identity Manager for IdentityIQ Integration

The Microsoft Forefront Identity Manager for IdentityIQ integration has dependencies on the following components:

- IQService
- Microsoft® Forefront® Identity Manager (FIM) 2010 Extensible Connectivity 2.0 Management Agent (ECMA 2.0)

IQService

You must install and register an IQService on windows host computer before provision to Microsoft® Forefront® Identity Manager. The IQService is a native Windows service that enables IdentityIQ to participate in a Windows environment and access information from Microsoft® Forefront® Identity Manager Synchronization Engine.

For more information on installation of IQService, see "Appendix F: IQService" of the *SailPoint IdentityIQ Direct Connectors Administration and Configuration Guide*.

> **Note:** **- Do not install IQService on the Windows Computer where Microsoft Forefront Synchronization Service is running.**
> **- The IQService ECMA Port should be allowed to communicate through fire wall on IQService installed host.**

To enable provisioning using IQService in Microsoft® Forefront® Identity Manager, perform the following:

1. Download `iiqIntegration-FIM-SyncEngineComponents.zip` from IdentityIQ installable zip files `$INSTALLDIR/integration/FIM`. Copy the `iiqIntegration-FIM-SyncEngineComponents.zip` file to a computer where IQService is installed.

2. Unzip the `iiqIntegration-FIM-SyncEngineComponents.zip` file and find `App.config` file located in the zip file.

3. Copy the `App.config` file to the IQService home directory.

4. Rename `App.config` file to `IQService.exe.config`.

Microsoft Forefront Identity Manager 2010 Extensible Connectivity 2.0 Management Agent

Using Microsoft Forefront Identity Manager Synchronization Engine Client create Extensible Connectivity 2.0 Management Agent using the `.dll` library file provided by SailPoint Technologies.

> **Note:** **Before creating Microsoft Forefront Identity Manager Extensible Connectivity 2.0 Management Agent, create the registry key** `HKEY_LOCAL_MACHINE\SOFTWARE\SailPoint\ECMA`**. This registry key is useful to log the Extensible Connectivity 2.0 Management Agent debug information.**

To create the registry key and management agent, perform the following:

1. Download `iiqIntegration-FIM-SyncEngineComponents.zip` from IdentityIQ installable zip files `$INSTALLDIR/integration/FIM`. Copy the `iiqIntegration-FIM-SyncEngineComponents.zip` file to a computer where FIM is installed under a temporary (*temp*) directory.

2. Unzip the `iiqIntegration-FIM-SyncEngineComponents.zip` file in the *temp* directory. The *temp* directory will have the following directory structure:

   – `temp\Config`

   – `temp\FimMA\Fim2010MA`

   – `temp\FimMA\Fim2010R2MA`

3. Create the registry key as follows:
   a. Open the Registry Editor. Import the `SailPointECMARegistry` file provided in the `temp\Config` file of the Registry Editor.
   b. After import, `HKEY_LOCAL_MACHINE\SOFTWARE\SailPoint\ECMA` registry key is created in the registry.

The registry contains the following default values:

- tracefile - C:\ecma_trace.log (Name and location of log file)

- tracelevel - 0 (Trace level for logging information)

To enable logging debug information for Extensible Connectivity 2.0 Management Agent, set the
trace level to **3** and to disable logging, set the trace level to **0**.

4. Copy the `.dll` library files provided in the `temp\FimMA\<FIM_Version>MA` to the following directory,
   where `FIM_Version` is the installed version of Microsoft Forefront Identity Manager:
   `FIM_HOME\2010\Synchronization Service\Extensions\`

5. Copy the `sailpoint.ecma.bindings.config` and `sailpoint.ecma.client.config` files provided
   in `temp\Config` directory to the following directory:
   `FIM_HOME\2010\Synchronization Service\bin\`

   Note: **To perform modification of the `sailpoint.ecma.client.config` and
   `sailpoint.ecma.bindings.config` files, provide the write permission to the 'Users' group.**

6. Stop FIM Synchronization Service.

7. Navigate to FIM_HOME\2010\Synchronization Service\bin\ directory and add the following configuration
   to `miiserver.exe.config` file under the <configuration> section:
   ```
   <!-- SailPoint Management Agent(ECMA2.0) Configuration -->
   <system.serviceModel>
       <bindings configSource="sailpoint.ecma.bindings.config"/>
       <client configSource="sailpoint.ecma.client.config"/>
   </system.serviceModel>
   ```

8. Start FIM Synchronization Service.

9. Using Microsoft Forefront Identity Manager Client GUI, import Management Agent using
   `temp\Config\SailPoint-MA.xml` configuration file.
   This will create the Management Agent with default Run profiles.

   Note: **Do not change the name and properties of the default Run profiles for above created Extensible
   Connectivity 2.0 Management Agent (SailPoint-MA).**

   Note: **To provisioned Metaverse Persons/Groups to newly created ECMA 2.0 Management Agent (that
   is, SailPoint-MA), use Metaverse Rules Extension or Synchronization Rules.**

Configuring IdentityIQ for Forefront Identity Manager Integration

This section describes the following:

- Create an IdentityIQ application of type Microsoft Forefront Identity Manager
- Run Microsoft Forefront Identity Manager Application Creator Task
- Aggregation from Microsoft Forefront Identity Manager Provisioning Integration Module

Create Forefront Identity Manager application in IdentityIQ

1. Create a new IdentityIQ application of type Microsoft Forefront Identity Manager. Enter the following
   required attributes of Forefront Identity Manager:

> **Note:** The application name for IdentityIQ application of type Microsoft Forefront Identity Manager must be different than that of any management agent's name in FIM server side.

Attributes	Description
IQService Host	Hostname of the computer where IQService is installed.
IQService Port	Port on which IQService is listening for IdentityIQ.
IQService ECMA Port	This port is opened by IQService to listen with Extensible Connectivity 2.0 Management Agent created in Microsoft Forefront Identity Manger Synchronization Engine.
User	Forefront Identity Manager user who has Administrator rights and must be a member of the following security groups on Microsoft Forefront Identity Manger Host: • FIMSyncBrowse • FimSyncAdmins User must belong to FIM Web Service (FIM portal) "Administrators" set. **Note: The name of the User can be in UPN (user@domain) or NetBIOS (domain/user) format.**
Password	Password of Forefront Identity Manager user.
FIM Service Host	Hostname or IP of the computer where Microsoft Forefront Identity Manager is installed.
FIM Service Port	Port for the Microsoft Forefront Identity Manager Service.
FIM Synchronization Service Host	Enter the host name or IP of the computer where Microsoft Forefront Identity Manager Synchronization Service is installed. **Note: This attribute is not required, if Synchronization service and Web Service are installed on the same host machine.**
FIM MA Name	Name of Extensible Connectivity 2.0 Management Agent created in Microsoft Forefront Identity Manger Synchronization Engine using library (**.dll**) provided by SailPoint Technologies.

2. Click **Test Connection** to verify the connection to Microsoft Forefront Identity Manager.

> **Note:** You can make use of the "FIM Application creator" task to discover all the application present in FIM environment. The input for this task would be newly created application of type "Microsoft Forefront Identity Manager" and executing this task would result in the creation of all multiplexed applications.

Run Microsoft Forefront Identity Manager Application Creator Task

This task creates an IdentityIQ application for each target system in Microsoft Forefront Identity Manager.

1. Select the Microsoft Forefront Identity Manager application created in step 1. on page 71.

2. Specify **Native Object Types of Account** this is Object type which is referred as account on native managed system (for example, Account, Person, User).

3. Specify **Native Object Types of Group** this is Object type which is referred as group on native managed system (for example, groups, Groups).

4.   Click **Save** and **Execute**.

Operation specific configuration on Microsoft Forefront Identity Manger

To communicate with Microsoft Forefront Synchronization Service, IQService uses Windows Management Instrumentation (WMI). Hence, enure that Windows Firewall on Microsoft Forefront Identity Manger Host should allow Windows Management Instrumentation (WMI) traffic.

This section describes the various configurations required for the following operations:

- Aggregation
- Provisioning

Aggregation from Microsoft Forefront Identity Manager Provisioning Integration Module

To aggregate Microsoft Forefront Identity Manager Persons/Groups and target Management Agent accounts/groups, create and execute an IdentityIQ Account Aggregation task for Application of type Microsoft Forefront Identity Manager.

Once aggregation is complete, all Persons/Groups from Microsoft Forefront Identity Manger Metaverse are created as accounts/groups in IdentityIQ and gets associated with IdentityIQ Application of type Microsoft Forefront Identity Manager. Additionally, all of the target Management Agent accounts/groups are aggregated automatically and associated with the respective child applications in IdentityIQ.

To support paging in IdentityIQ Aggregation task, Extensible Connectivity 2.0 Management Agent library supports paging in Export, Full Export run profiles. Maximum batch size supported by Extensible Connectivity 2.0 Management Agent library is 500. The default batch size provided is 100.

Configuration for Aggregation

In Aggregation, SailPoint IdentityIQ does not read users and groups from Microsoft Forefront Identity Manger Metaverse. Instead, it reads users and groups from newly created ECMA 2.0 Management Agent (that is, SailPoint-MA). Before starting aggregation from SailPoint IdentityIQ, provision Microsoft Forefront Identity Manger Metaverse Persons/Groups to newly created ECMA 2.0 Management Agent (that is, SailPoint-MA).

To provision Metaverse Persons/Groups to newly created SailPoint Management Agent (ECMA 2.0), use Metaverse Rules Extension or Synchronization Rules.

Provisioning from Microsoft Forefront Identity Manager Provisioning Integration Module

In provisioning, SailPoint IdentityIQ pushes users/groups data only to newly created SailPoint Management Agent (ECMA 2.0) connector space.

Configuration for Provisioning

To project SailPoint Management Agent connector space data changes to Microsoft Forefront Identity Manger Metaverse, use the following configuration rules for newly created SailPoint Management Agent (ECMA 2.0) as per their sites specifications:

- Management Agent Rules Extensions
- Mapping rules (Attribute Flow Rules)
- Join Rules
- Projection Rules
- Deprovisioning Rules

Provisioning examples

This section describes the various examples of provisioning.

- **Create Account for Microsoft Forefront Identity Manager target Management Agent Active Directory**: This operation includes the following steps:

  a. IdentityIQ sends the **Create Account Request** for FIM Active Directory Management Agent.

     Request is received by newly created SailPoint Management Agent (ECMA 2.0) on FIM side.

  b. SailPoint Management Agent (ECMA 2.0) adds new account in its own connector space.

  c. Microsoft Forefront Identity Manger Synchronization Engine projects these connector space changes to metaverse using configuration rules (rules listed in "Configuration for Provisioning" section).

     In Metaverse, new Person is created and link between Metaverse person and SailPoint Management Agent (ECMA.2.0) Account is created.

  d. To provision new Metaverse Person to Active Directory Management Agent connector space, Synchronization Engine uses Metaverse rules Extension or Synchronization rule. If rule condition is satisfied, Synchronization engine provisions Metaverse Person to Active Directory Management Agent connector space.

     Note: Depending on the Metaverse rules Extension or Synchronization rule configured in Synchronization Engine, Account on other Management Agent (native system) will be created.

  e. Synchronization Engine creates link between Metaverse Person and Active Directory Management Agent connector space account.

  f. After running **Export** run profile on FIM Active Directory Management Agent, the Active Directory

Management Agent connector space's account gets created on native system.

- **Enable Account for Microsoft Forefront Identity Manger target Management Agent Active Directory**: Enable Account operation from IdentityIQ will set the value for SailPoint Management Agents **enableStatus** attribute to **True**. The **enableStatus** attribute can be mapped to user defined Metaverse attribute for propagating the changes to other interested Management Agent's connector space attributes. After running **Export** run-profile on interested management agents, status for accounts will be enabled.

- **Disable Account for Microsoft Forefront Identity Manger target Management Agent Active Directory**: Disable Account operation from IdentityIQ will set the value for SailPoint Management Agents **enableStatus** attribute to **False**. The **enableStatus** attribute can be mapped to user defined Metaverse attribute for propagating the changes to other interested Management Agent's connector space attributes. After running **Export** run-profile on interested management agents, status for accounts will be disabled.

- **Unlock Account for Microsoft Forefront Identity Manger target Management Agent Active Directory**: Unlock Account operation from IdentityIQ will set the value for SailPoint Management Agents **lockStatus** attribute to **False**. The **lockStatus** attribute can be mapped to user defined Metaverse attribute for propagating the changes to other interested Management Agent's connector space attributes. After running **Export** run-profile on interested management agents, status for accounts will be unlocked.

- **Change/Reset Account password for Microsoft Forefront Identity Manger target Management Agents**: Change Account password operation from IdentityIQ will set the value for SailPoint Management Agents **password** attribute. The **password** attribute can be mapped to user defined Metaverse attribute for password. The metaverse password attribute should be mapped in metaverse outbound mappings for target management agents **export_password** attribute. After running **Export** run-profile on interested management agents, password gets changed on native system's accounts.

Troubleshooting

This section provides the resolutions for the following errors that may be encountered while setting up and configuring Microsoft Forefront Identity Manager.

1 - An unexpected error occurred: sailpoint.connector.ConnectorException: Errors returned from IQService. The maximum message size quota for incoming messages (3310720) has been exceeded. To increase the quota, use the MaxReceivedMessageSize property on the appropriate binding element.

When running any task (aggregation, application generation, and so on) related to Microsoft Forefront Identity Manager Application, the following error message may appear:

```
An unexpected error occurred: sailpoint.connector.ConnectorException: Errors returned
from IQService. The maximum message size quota for incoming messages (3310720) has
been exceeded. To increase the quota, use the MaxReceivedMessageSize property on the
appropriate binding element.
```

Resolution: Perform the following:

1. Increase the size of **MaxReceivedMessageSize** parameter as required, from the `IQService.exe.config` file and then restart IQservice.

2. If similar type of error is observed for other parameters then increase the size of particular parameter from the `IQService.exe.config` file and then restart IQservice.

2 - An unsecured or incorrectly secured fault was received from the other party. See the inner FaultException for the fault code and detail. An error occurred when verifying security for the message."

When running any task (aggregation, application generation, and so on) related to Microsoft Forefront Identity manager application, the following error message may appear:

```
An unsecured or incorrectly secured fault was received from the other party. See the
inner FaultException for the fault code and detail. An error occurred when verifying
security for the message.
```

Resolution: Perform the following:

1. Verify date and time on the host computer where IQService is running.

2. Verify date and time on the host computer where Forefront Identity Manager is running.

3. Ensure that, both host do not have time difference more than 10 minute.

3 - While creating Extensible Connectivity 2.0 Management Agent (SailPoint-MA), the following error may be observed in ECMA log file:

```
System.ServiceModel.QuotaExceededException: The maximum message size quota for
incoming messages (65535) has been exceeded. To increase the quota, use the
MaxReceivedMessageSize property on the appropriate binding element.
```

Resolution: Perform the following:

1. Increase the size of **MaxReceivedMessageSize** parameter as required from the `sailpoint.ecma.bind-ings.config` file and restart FIM Synchronization Service.

2. If similar type of error is observed for other parameters then increase the size of particular parameter from the `sailpoint.ecma.bindings.config` file and then restart FIM Synchronization Service.

Known/Open issue

Following are the known/open issues of Microsoft Forefront Identity Manager:

- User updates to attributes that affect target system accounts require a synchronization rule to be setup and run on the FIM server for the change to propagate.

- Due to Microsoft Forefront Identity Manager Synchronization's engine behavior, the Provisioning Integration Module will process all the request sequentially.

- Refresh schema action on SailPoint provided Extensible Connectivity 2.0 Management Agent (ECMA2.0) library is not supported.

- Delta Aggregation limitations

  - If new delta data is available in the SailPoint Management Agent (SailPoint-MA) connector space and user is unaware of it then there is a possibility that user might run Provisioning operation on the new delta data before running delta aggregation. If the new delta data is Provisioned before delta aggregation then the status of all the new delta data will be changed to **Awaiting Export**. In this scenario, running delta aggregation will not work since the status delta data is not **Pending Export**.

  - At a given time, only account aggregation or group aggregation is supported in a sequence such that account aggregation must be run first.

Known/Open issue

Assisted Deployment Integration Modules:
Service Management Integration Modules

IdentityIQ supports Service Management Systems for Service Desk Integration and Service Catalog Integration.

Service Desk Integration

Creates tickets in Helpdesk systems based on compliance and provisioning actions taken in IdentityIQ (For example, new account request).

- "SailPoint HP Service Manager Service Integration Module" on page 81

Service Catalog Integration

Enables enterprises to include LCM requests (for example, access request, password change) in the service catalog of the Service Request Management system.

- "SailPoint ServiceNow Service Catalog Integration" on page 95

> **Note:** **A minority of SailPoint customers have deployed the Integration Modules in this section. SailPoint will provide assistance during the deployment of these integrations. Additional troubleshooting, diagnostic, and best practice information beyond what is contained in this document will be provided on Compass, SailPoint's Online customer portal. In some instances, SailPoint will guide the deployment team and actively participate in the design, configuration, and testing of the integration to the managed system.**
> **For more specific information, refer to the Connector and Integration Deployment Center on Compass.**

Chapter 10: SailPoint HP Service Manager Service Integration Module

The following topics are discussed in this chapter:

Overview

> **Note:** **SailPoint will provide assistance during the deployment of this Integration Module. Additional troubleshooting, diagnostic, and best practice information beyond what is contained in this document will be provided in the Connector and Integration Deployment Center on Compass.**

The integration between SailPoint and HP Service Manager Desk enables customers to create incidents and change requests in HP Service Manager for the configured operations (for example, Change Password, Request Entitlement and so on) for the configured application. The seamless integration of SailPoint and HP Service Manager Service Integration Module (HP SIM) eliminates the need to build and maintain a custom integration, and speeds time-to-deployment.

Supported features

HP Service Manager Service Integration Module supports the following features:

- Creates Incidents and Change Requests in HP Service Manger platform through provisioning request in IdentityIQ.
- Fetching the status of Incident from HP Service Manager and update the status of the respective Access Requests in IdentityIQ.
- Fetching the status of Change Request from HP Service Manager and update the status of the respective Access Requests in IdentityIQ.
- Retry Mechanism for Create Ticket request failure

Supported platforms

SailPoint HP Service Manager Service Integration Module supports the following version of Service Manager:

- HP Service Manager version 9.3

Pre-requisites

- Ensure that HP Service Manager 9.3 is installed and running.
- Ensure that the HP Service Manager 9.3 Server service is started.
- When running a HP integration on an application server using Java 6, the JRE must be directed to use the SOAP/SAAJ implementation provided by Axis2. To do this, add the following Java options to the application server:

  - `-Djavax.xml.soap.SOAPConnectionFactory=org.apache.axis2.saaj.SOAPConnectionFactoryImpl`

  - `-Djavax.xml.soap.MessageFactory=org.apache.axis2.saaj.MessageFactoryImpl`

  - `-Djavax.xml.soap.SOAPFactory=org.apache.axis2.saaj.SOAPFactoryImpl`

  Without these options, a `ClassCastException` error or Signature creation failed error will occur when the HP integration is used.

- Ensure that the HP Service Manager Service Integration Module is up and running: Ensure that the following WSDL is accessible:

  - For incident: **http://<host>:<port>/SM/7/IncidentManagement.wsdl**

  - For change request: **http://<host>:<port>/SM/7/ChangeManagement.wsdl**

  Where <host> is the host name of the system where HP Service Manager is setup and <port> is port number configured for the above web services on HP Service Manager setup.Alternatively, use a Soap UI tool to submit a simple request (for example, incident). For convenience you can use the basic authentication mechanism for authorization with SOAP UI tool to confirm that the web service layer is functional.

Permission for HP Service Manager User

To obtain the minimum permission required for a HP Service Manager User, perform the following:

1. Create a new contact as follows:

   - Navigate to **System Administration ==> Base System Configuration ==> Scheduled Maintenance ==> Contacts**

Enter **Contact Name** and **Full Name**. Click on **Add** button as displayed in the following figure:

Pre-requisites
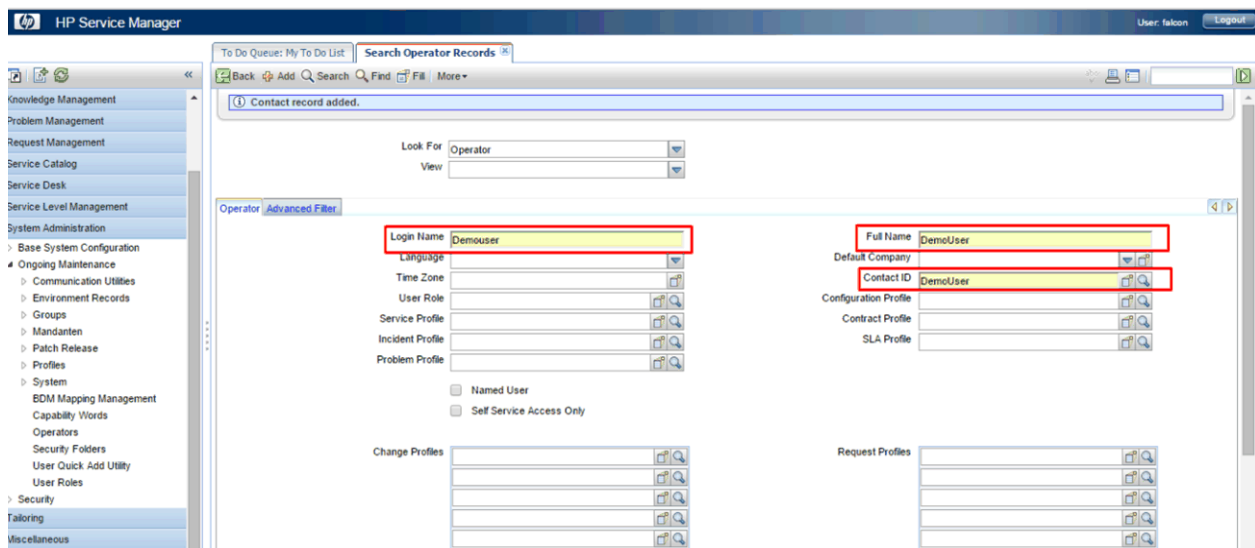
2. Create new operators as follows:

   - Navigate to **System Administration ==> Ongoing Maintenance ==> Operators**

   - Enter the following details and click on **Add**.

     • Login Name

     • Full Name

     • Contact ID

     • Default Company

3. Create new application profiles as follows:
   On the Operator Record page, under the Application Profiles tab, enter the details of the following parameters (as shown in the following figure):

Parameter	Value
User Role	System Administrator
Service Profile	sysadmin
Incident Profile	sysadmin
Problem Profile	sysadmin
Configuration Profile	sysadmin
Contract Profile	sysadmin
SLA Profile	sysadmin
Change Profiles	sysadmin
Request Profiles	sysadmin
Security Roles	System Administrator

4. On the Operator Record page, under the Security tab, enter the **Password** and select the **Unlimited Sessions** and **Prevent Lockout** check boxes.

5. Create new login profiles as follows:
    On the Operator Record page, under the Login Profiles tab, enter the details of the following parameters (as shown in the following figure) and select the **Named User** check box:

    - Date Format: mm/dd/yy

    - Message Level: Information

6. On the Operator Record page, under the Startup tab, enter the details of the following parameters (as shown in the following figure) and select the **Activate Command Line on Startup** checkbox:

Parameters	Value
RAD Name	menu.manager
name	MAIN MENU
prompt	

Parameters	Value
string1	HOME

In the above figure select the values for Executive Capabilities and Query Groups from the drop down list as follows and click the **Save** button:

- Execute Capabilities

  - partial.key

  - SysAdmin

  - SQLAdmin

  - SOAP API

  - user.favorites

- Query Groups

  - Service Desk

  - Interactions

Configuring HP Service Manager for IdentityIQ Integration

This section provides the required information for configuring IdentityIQ to integrate with HP Service Manager. This integration enables IdentityIQ to create tickets for requested revocations, track ticket numbers in association with revocation tasks, and update IdentityIQ with the status of current tickets.

SailPoint provides a default HP Service Manager Service Integration configuration. This configuration implements the integration between IdentityIQ and the HP to fulfill creation of tickets based on IdentityIQ access certification remediation events.

When integrating with **changeRequest**, modify `HPServiceManagerIntegrationConfigForChange.xml` sample file and import in IdentityIQ. When integrating with **incident**, modify `HPServiceManagerIntegrationConfigForIncident.xml` file and import in IdentityIQ.

The default configuration is located in *iiqHome*`/WEB-INF/config/` directory, where *iiqHome* is the location where IdentityIQ was installed.

This section explains the various entries that are specific for this integration. For more information of the entries in the `IntegrationConfig` file, see Appendix: A: Common Identity Management Integration Configuration.

The integration configuration must include the following entries:
- **endpoint**: URL to the web service
- **namespace**: namespace of the XML returned by the web service
- **prefix**: prefix associated with the namespace

The integration configuration includes the following entries if the web service side of the integration is configured for authentication using the SOAP authentication specifications:
- username
- password
- statusMap

The web services and authentication entries are consumed by configuration entries for each web service. They can be positioned either within the configuration entries themselves or as children of the Attributes element. Entries that are children of the Attributes element can be thought of as global values, while entries within the configuration entities can be thought of as local.

For example, if both entries share the same authentication credentials, those credentials might be placed in the Attributes element as peers of the configuration entries and the integration code searches the parent entry for the credentials if they are not found in the configuration entries. Conversely, if the configuration entries have different endpoints (are handled by separate web services), each configuration entry specifies the endpoint of the web service to call and any value outside of the configuration entry is ignored.

There are two supported configuration entries for integration with HP Service Manager. These entries are children of the integration Attributes element:
- getRequestStatus
- provision

The values of each are Map elements containing key/value pairings of the configuration data. They contain the specific data needed by the **getRequestStatus()** and **provision()** methods of the IdentityIQ integration executor and correspond to HP Service Manager Web Service methods.

The **getRequestStatus** and **provision** entries contain the following entries:
- **soapMessage** (required): full XML template of the entire SOAP envelope that is sent to the web service. The integration code first runs this template through Apache's Velocity template engine to provide the data needed by the web service.
- **responseElement** (required): name of the element containing the results of the web service call (for example, the element containing the ticket number opened by the web service in response to the call from IdentityIQ).
- **SOAPAction** (required): SOAP request's action.
- **endpoint** (required)
- **namespace** (required)

- **prefix** (required)

Before a template is sent to the web service, it is processed by the **Velocity template engine**. The integration code provides different data objects to Velocity for evaluation based on the integration method.

The **provision** call passes the following objects to Velocity:

- **config**: the integration configuration for provision, represented as a Map
- **provisioningPlan**: the data model of the provision request

The **getRequestStatus** call passes the following objects to Velocity:

- **config**: the integration configuration for getRequestStatus, represented as a Map
- **requestID**: the string ID of the request whose status is being queried

Both calls have access to a timestamp variable containing a current Date object and a dateFormatter object. The dateFormatter is built using an optional dateFormat attribute from the **config** object. If the dateFormat attribute does not exist, the formatter defaults to the pattern EEE, d MMM yyyy HH:mm:ss z.

Sample XML files for Incident and Change Request

The entries contained in the Map are the only required entries. Any authentication information required by this integration is inherited from the parent Attributes element.

For more information and examples of the sample files, see the following sample files:

- HPServiceManagerIntegrationConfigForIncident.xml
- HPServiceManagerIntegrationConfigForChange.xml

If any changes required in the mapping, change the value/key values in "statusMap" and "statusMapCloserCode" as mentioned in the following tables for Incident and Change Request:

Incident Request

Entry key	Values
statusMap	
Closed	committed
Pending Other	inProcess
Referred	inProcess
Replaced Problem	inProcess
Resolved	committed
Open	inProcess
Accepted	inProcess
Rejected	inProcess
Work In Progress	inProcess
Pending Customer	inProcess
Pending Vendor	inProcess
Pending Change	inProcess
statusMapCloserCode	
Automatically Closed	committed
Not Reproducible	committed
Out of Scope	committed
Request Rejected	committed
Solved by Change/Service Request	committed
Solved by User Instruction	committed
Solved by Workaround	committed
Unable to solve	failure
Withdrawn by User	failure

Change Request

Entry Key	Values
statusMap	
initial	inProcess
waiting	inProcess
reopened	inProcess
closed	committed
statusMapCloserCode	
1 - Successful	committed

Entry Key	Values
2 - Successful (with problems)	committed
3 - Failed	failure
4 - Rejected	failure
5 - Withdrawn	failure
6 - Cancelled	failure

Configuration procedure

The following steps should be performed to modify the default HP Service Manager Service Integration configuration for a specific HP Service Manager Server.

1. Obtain the environment-specific Web Service "endpoint", for example, **http://<host>:<port>/SM/7/ws**.

2. Once you are familiar with the WSDL, modify the default IdentityIQ HP Service Manager configuration using the information collected about the web service.

   a. In the <IntegrationConfig> element of the integration configuration, modify the **username** and **password** entries in the attributes map to contain the credentials required for authentication to the web service.

   b. In the <IntegrationConfig> element of the integration configuration, modify the provision entry of the Attributes map by setting the endpoint, and, if necessary, the namespace, the prefix, the responseElement, and the soapMessage attributes (the default values: IdentityIQ HP Service Manager IntegrationConfig):

      i. Set the value for endpoint to the value located in the WSDL earlier.

      Note: **The value in the IdentityIQ integration configuration must be a valid HTTP URL and have any special characters escaped. The most common change that must be made is to replace all and symbols with &**

      ii. The value for namespace comes from the **targetNamespace** attribute of the **xsd:schema** element in the WSDL.

      iii. The value for prefix is the prefix of the XML elements that will be contained in the SOAP response.

      iv. The value for **responseElement** should be the HP Service Manager form field that corresponds to the id of the form that the web service creates.

      v. The value for **soapMessage** should be the SOAP message body that IdentityIQ will send to HP Service Manager. The exact format of this message is a function of the form that is published as described by the form's WSDL. The XML elements in the **soapenv:Body** element should be changed to match the HP Service Manager form fields for the published web service. Each required HP Service Manager form field must have an element in the SOAP message. The value can be fixed or can be a variable that will be substituted using IdentityIQ's Velocity templating

The information in the reference section above show the variables that are provided and the example integration configuration provides examples of how they are used.

Retryable mechanism

By default HP Service Manager Service Integration Module provides retry mechanism for Connection reset and for unknown host problems occurred from network issues.

However you can configure **retryableErrors** list in integration configuration (`IntegrationConfig`) file to add new exception strings to the attributes map in integration configuration file.

The **retryableErrors** entry is a list of strings through which the integration searches when it receives a message from the Service Integration Module. Only **SOAPException** strings are considered for retry that is, the exceptions raised from SOAP web service. If one of the strings in the entry exists in the error, the integration attempts to retry the request. When the configured error string is not a part of the error message returned from the Service Integration Module, then IdentityIQ will not attempt a retry.

For example,

```
<entry key="retryableErrors">

    <value>

      <List>

        <String>Connection reset</String>

      </List>

    </value>

</entry>
```

> **Note:** Error messages containing very specific information about date/time, sequence ID and so on must be avoided. Error codes or error message substrings would be good candidates for inclusion. Only exceptions raised from soap web service are considered for retry.

Troubleshooting

This section provides the resolutions for the following errors that may be encountered while setting up and configuring HP Service Manager Service Integration Module.

1 - 'Authorization Required' error messages

The following type of error messages appear when the authorization data is not sent to HP Service Manager:

```
Caused by: org.apache.axis2.AxisFault: Transport error: 401 Error:
Authorization Required
```

Resolution: Verify the procedure to configure appropriate authorization mechanism. For more information on the procedure, see "Configuration procedure" on page 91.

2 - Document Type Declaration (DTD) parsing errors

The DTD parsing errors appear when the following JVM arguments are not defined for your application server:

- `-Djavax.xml.soap.SOAPConnectionFactory=org.apache.axis2.saaj.SOAPConnectionFactoryImpl`
- `-Djavax.xml.soap.MessageFactory=org.apache.axis2.saaj.MessageFactoryImpl`
- `-Djavax.xml.soap.SOAPFactory=org.apache.axis2.saaj.SOAPFactoryImpl`

Resolution: Ensure that the application is pointing to the correct java runtime (that is, 1.6) and the above mentioned JVM arguments are defined for application server.

Chapter 11: SailPoint ServiceNow Service Catalog Integration

The following topics are discussed in this chapter:

Overview

> **Note:** **SailPoint will provide assistance during the deployment of this Integration Module. Additional troubleshooting, diagnostic, and best practice information beyond what is contained in this document will be provided in the Connector and Integration Deployment Center on Compass.**

SailPoint Service Catalog Integration is an integration between ServiceNow and SailPoint IdentityIQ. This allows users of both systems to easily navigate from ServiceNow into IdentityIQ, and gives users a "one stop shop" to request all IT related items.

The integration between SailPoint and ServiceNow gives mutual customers a complementary identity access governance and service management solution that works together to ensure strong controls are in place to meet ever stringent security and compliance requirements around user access to sensitive applications. The integration also allows users to perform other activities (such as password changes, approve access, manage account) that are configured within the system.

The chapter describe the implementation approach and configuration of the SailPoint Service Catalog Integration in ServiceNow. The following information enables administrators to deploy and maintain the integration.

The flow of SailPoint Service Catalog Integration in ServiceNow is as follows:

- User logs in to ServiceNow.
- The **SailPoint Service Catalog Integration** application is available to the logged in user.
- Click on any of the links mentioned in the request section of SailPoint Service Catalog Integration Application. Request flows to IdentityIQ and IdentityIQ page is opened in ServiceNow.
- Complete the request in IdentityIQ page opened in ServiceNow and Ticket would be generated in ServiceNow for the same request. Initially the status of ServiceNow ticket would be **open** or **openNoApproval**.
- Once the request is approved in IdentityIQ, ServiceNow ticket's status would be updated.
- Once the provisioning of request is done on IdentityIQ side, the ticket's status would be updated (closed) in ServiceNow.

The following diagram represents the high level flow diagram for Service Catalog Integration with SailPoint IdentiyIQ:
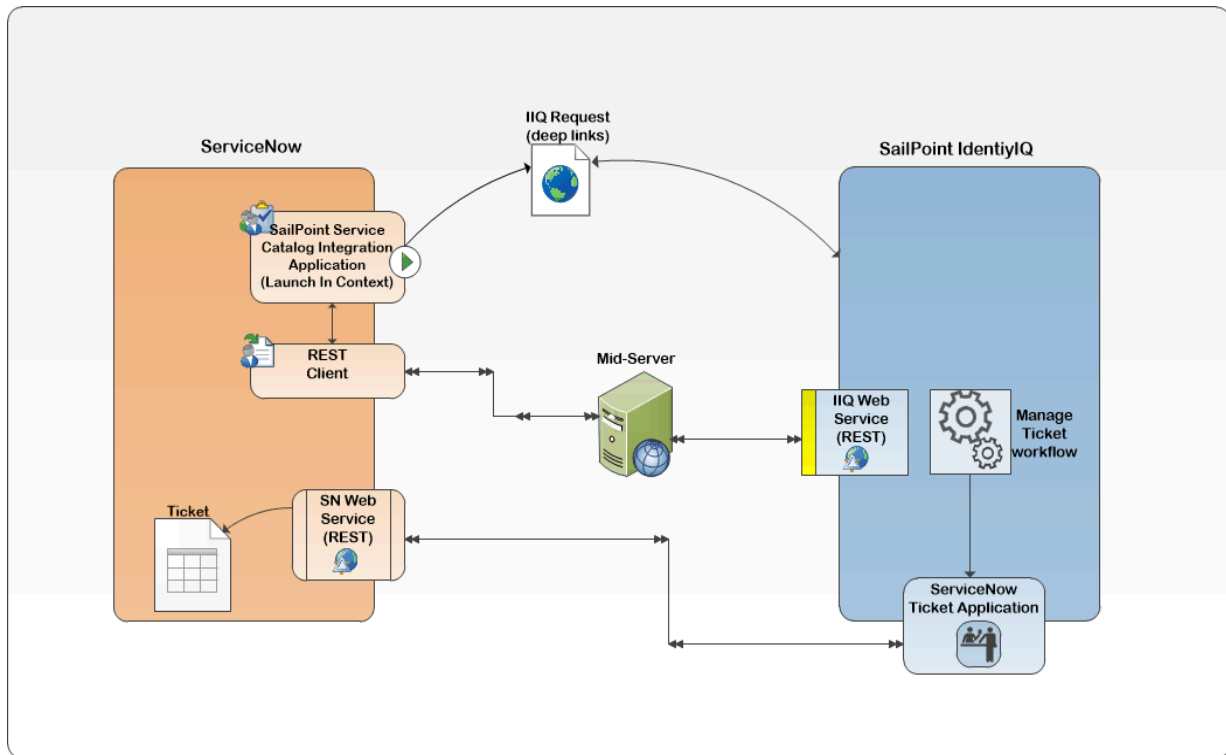
Figure 1—Basic Configuration

If a request is submitted from ServiceNow then SailPoint IdentityIQ creates or updates tickets in ServiceNow using ServiceNow Ticket Application from IdentityIQ.

Supported features

The SailPoint Service Catalog Integration enables the use of IdentityIQ Lifecycle Manager within ServiceNow user interfaces. The inclusion of the SailPoint Service Catalog Integration Application provides the following functionality within ServiceNow:

- Manage Account for Me

    - Enable/Disable/Unlock/Delete Accounts

- Request Access for Me/Others

    - Request for Entitlement

    - Request for Roles

- Request Password Changes
- My Access Approvals

- Track My Requests

    - Get Request Status

    - Get complete details of request Item

Supported platforms

SailPoint Service Catalog Integration supports the following ServiceNow instance versions:

- Geneva
- Fuji
- Eureka

   Note: **Minimum version for Fuji must be "Fuji with patch 4 or higher".**

Pre-requisites

- For the integration to successfully authenticate the user, the ServiceNow accounts (users) must have correlated identities in IdentityIQ.
- ServiceNow Mid Server must be installed. See "Mid server installation" on page 99.

   Note: **Ensure that the WebServer hosting IdentityIQ must be SSL enabled.**

Installation and configuration in ServiceNow

This section describes the installation and configuration procedures for SailPoint IdentityIQ ServiceNow Service Catalog Integration.

Installation

This section describes the installation procedure.

Install update set in ServiceNow

Apply the IdentityIQ ServiceNow Service Catalog Integration update set:

1. Copy the relevant update set from the following directory to a temporary directory:
   `identityiq-`*`releaseVersion`*`.zip\integration\servicenow\iiqIntegration-ServiceNow.z ip\ServiceCatalogUpdateSet\IdentityIQServiceNowServiceCatalog.v`*`x.x_serviceNowIns tanceVersion`*`.xml`

In the above directory:

- *releaseVersion* is the version of the current IdentityIQ release

- *serviceNowInstanceVersion* is **geneva, fuji,** or **eureka**

- *x.x* is the version of the Catalog integration as described in the following table:

ServiceNow version	Update Sets
Geneva	IdentityIQServiceNowServiceCatalog.v1.0_geneva.xml
Fuji	IdentityIQServiceNowServiceCatalog.v1.2_fuji.xml
Eureka	IdentityIQServiceNowServiceCatalog.v2.0_eureka.xml

2. Import relevant update set in ServiceNow instance. For more information and guidelines on usage of the update set, refer to the following wiki link:
   http://wiki.servicenow.com/index.php?title=Saving_Customizations_in_a_Single_XML_File
   In the above link, refer to section 3 "Loading Customizations from a Single XML File".

   After successfully applying the update set in ServiceNow the **SailPoint Service Catalog Integration** application is created.

3. (*For Fuji or Later*) After committing the update set for SailPoint Service Catalog Integration for Fuji or Later, add read operation ACL's on **syslog_app_scope** and **sys_script_include** tables as mentioned in the following steps:
   a. Login to ServiceNow Fuji instance with administrator credentials.
   b. Ensure that, the **Global** application is selected.
   c. Create wildcard field rules (*) ACL for **syslog_app_scope** table for Read operation with required **x_sapo_iiq_catalog.spnt_admin** role.
   d. Create wildcard field rules (*) ACL for **sys_script_include** table for Read operation with required **x_sapo_iiq_catalog.spnt_admin** role.

Overview "SailPoint IdentityIQ ServiceNow Service Catalog Integration" Application

The SailPoint IdentityIQ ServiceNow Service Catalog Integration provides easy access to IdentityIQ Services, Access Requests and Access Approvals.

> **Note:** **The SailPoint Service Catalog Integration application is available only to those users who have the following role assigned:**
> - x_sapo_iiq_catalog.spnt_user **for Fuji or Later instances**
> - spnt_identityiq_user **for Eureka instance.**

1. My Access Requests

   - Displays a list of access requests which include the Access Requests Opened By or Requests for the current user logged in.

   - By clicking into the IdentityIQ request, a user can see the Request information, as well as all of the Request Items with a brief description of what the Request Item entailed.

   - By clicking on the **View Detail** button a user can view the details of this Request in IdentityIQ.

2. My Access Approvals

   - Allows user to view the pending approvals in IdentityIQ for the logged in user. In addition to this, the module shows number of pending approvals for the logged in user with count next to link.

   - The **My Access Approvals** link does not display the number of pending approvals on ServiceNow Geneva version.

3. Request Access for Me

   - Allows user to request application access.

   - The **Request Access for Me** directly links to the request access page, allowing currently logged in user to request entitlements and roles.

4. Request Access for Others

   - The **Request Access for Others** link displays a dialog box so that the current user can select a different user to request access on their behalf. Once a user is selected, this links to the request access page.

   Note: **The "Request Access for Others" requires either the ServiceNow `admin` or the following role to display:**
   - (*For Fuji or Later*) `x_sapo_iiq_catalog.spnt_manager`
   - (*For Eureka*) `spnt_identityiq_manager`
   The above role is the custom role created to give to managers of other users from IdentityIQ in ServiceNow.

5. Change Password

   - Manages password for different accounts managed in IdentityIQ.

   - The **Change Password** link directs the link to the logged in users to manage the passwords in IdentityIQ.

6. Manage Accounts for Me

   - The **Manage Accounts for Me** opens the manage accounts page in IdentityIQ.

7. Track My Request

   - Allows user to check status regarding a recent access request.

   - Track My Requests opens the Access Request page in IdentityIQ.

8. Properties

   - Provides the form for logging in to the IdentityIQ API and performing a REST request to the MID SERVER and writing debug inform action to the Log.

9. Logs

   - Allows to view debugging, error and general information messages as they are written to the Log.

10. Customer Support

    - Allows the user to navigate to a support page located at Customer Support.

Mid server installation

For an overview and installation instructions on setting up the mid server, see the following ServiceNow wiki:

http://wiki.servicenow.com/index.php?title=MID_Server

Geneva version has an additional installation step of mid server validation. For more information, see the following ServiceNow wiki:

https://docs.servicenow.com/bundle/geneva-it-operations-management/page/product/mid_server/task/t_ValidateAMIDServer.html

Configuration

Set "SailPoint Service Catalog Integration" Application properties in ServiceNow to point to the SailPoint IdentityIQ instance

Open the **Properties** module from **SailPoint Service Catalog Integration** Application and modify the following properties:

> **Note:** The properties module from SailPoint Service Catalog Integration application is available only to those user who have the following role in ServiceNow:
> - (*For Fuji or Later*) `x_sapo_iiq_catalog.spnt_admin`
> - (*For Eureka*) `spnt_identityiq_admin`

- **Use Single-Sign On for IdentityIQ Requests**: Determines whether SailPoint IdentityIQ is setup for Single Sign On.

    - True: ServiceNow assumes no authentication is required when opening direct links into IdentityIQ.

    - False: ServiceNow uses the remoteLogin web service to retrieve a one-time use authentication token to login the current user.

- **IdentityIQ Instance Endpoint URL**: URL specifying the SailPoint IdentityIQ endpoint.

- **User to login to the IdentityIQ API**: Username for authentication during REST Requests to IdentityIQ.

    > **Note:** This user should have System Administrator Permission in IdentityIQ

- **Password to login to the IdentityIQ API**: Password for authenticating during REST Requests to IdentityIQ.

- **Mid Server to use to make REST requests to IdentityIQ**: The name of the Mid Server to make REST Requests through to IdentityIQ.

- **IdentityIQ Log Level**

    - (*For Fuji or Later*) The SailPoint Service Catalog Integration application's properties module contains the following new properties related to logger:

        - **Logging Level:** The default value for this property is **warn**.

        - **Logging Destination:** The default value for this property is **DB**.

            If **DB** is selected, then logs will be transformed to system tables.

            If **FILE** is selected, then logs will be transformed to system node files.

    - (*For Eureka*) Log level for the IdentityIQ Access Request application. By default the value of the **IdentityIQ Log Level** parameter is **debug**, for production environments it is suggested to set the value to **error**.

- **Interval in which to poll for approvals**: Identifies the interval in which the approvals on the server side are verified. Default: 15 minutes.
  If the IdentityIQ Approval record for the user logged in has not been updated within 15 minutes, the user is logged out to IdentityIQ to request an update of the number of approvals outstanding.

- **Application Name within IdentityIQ for the ServiceNow Integration**: Application Name within IdentityIQ for the ServiceNow Integration. Default: `SailPointServiceCatalog`
- **IdentityIQ ServiceNow Application Name**: ServiceNow users are aggregated in IdentityIQ through ServiceNow connector application.

Mid server Setup

The Mid Server must be setup with a polling time lower than normal. This allows the shortest lag time when loading IdentityIQ pages when ServiceNow must request a remoteLogin token.

MID Server poll time: Sets the MID Server polling interval (in seconds).
Type: integer (seconds)
Default value: 15
`mid.poll.time`

Note: **It is suggested to set the "MID Server poll time" property to 5 seconds, to increase the rate in which REST Message Requests will be picked up and processed.**

Note: **ServiceNow and IdentityIQ instance must be accessible to Mid-server.**

Configuration in SailPoint IdentityIQ

Perform the following procedure to create Application in IdentityIQ to manipulate Tickets in ServiceNow:

1. Create **SailPointServiceCatalog** ticketing application using default configuration file located in
   `iiqHome/WEB-INF/config/SailPointServiceCatalog.xml` directory.
   In the above directory, *iiqHome* is the location where IdentityIQ is installed.

   This creates **SailPointServiceCatalogIntegration** rule in IdentityIQ.

2. Modify the following parameters:

   - **url**: Service Now JSON API endpoint

     (*For Fuji or Later*) **https://<servicenow-base-url>/x_sapo_iiq_catalog_processor.do?action=POST**

     (*For Eureka*) **https://<servicenow-base-url>/iiq.do**

   - **username**: ServiceNow user who has System Administrators role

   - **password**: Password of an user

3. Modify the following parameter in the **SailPointServiceCatalogIntegration** rule:

   - **connectorAppName**: Provide the ServiceNow connector application name through which ServiceNow users are aggregated in IdentityIQ. The value for this parameter must be the name of the Connector Application pointing to ServiceNow instance where ServiceNow Catalog Integration is configured. Default value is **null**

   Note: - If the rule name is changed in Step 1. then change the value for `ticketDataGenerationRule` entry in `SailPointServiceCatalog.xml` file.
   - If Application Name is changed in SailPoint Identity Access Request Application properties then change name in `SailPointServiceCatalog.xml` file.

4. Enable **iFrame** option in IdentityIQ by performing the following steps:
   a. Navigate to IdentityIQ debug page.
   b. Select **System Configuration** from **Configuration Objects**.
   c. Find the key **allowiFrame** and set the value to **true.**

        d.   Save the **System Configuration.**

Troubleshooting

1 - Enable Traces for SailPoint Service Catalog Integration application in ServiceNow

Resolution: To enable the traces in SailPoint Service Catalog Integration application set the following property to **debug**:

- (*For Fuji or Later*) **Logging Level** (`x_sapo_iiq_catalog.logging.verbosity`)
- (*For Eureka*) **IdentityIQ Log Level** (`com.sailpoint.iiq.log_level`)

2 - Enable Stack Tracing for IdentityIQ

Resolution: Set the workflow **Trace** attribute to **true** when configuring the ServiceNow Service Catalog Integration application parameters to enable logging. This enables any IdentityIQ provisioning actions traceable.

Assisted Deployment Integration Modules:
Mobile Device Management Integration Modules

Mobile Device Management Integration Module (MIM) manages Devices enrolled in Mobile Device Management (MDM) System. MIM also manages Users/Administrators of MDM System. Following is an architecture diagram of MIM:
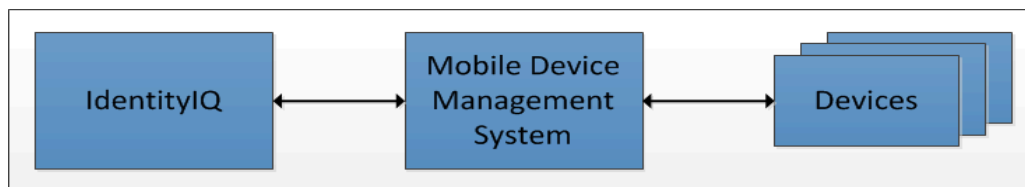
MDM systems import users from a central directory server or maintains its own repository. Operations to be performed on devices are sent to the respective MDM System which then performs the specified action on the target device. MIM does not communicate with the devices directly.

MIM uses two separate applications to manage users and devices. Main MIM application manages users in the MDM System and the proxy MIM application manages devices in the MDM System. In some cases there is one application which manages devices.

This section contains information on the following:

- "SailPoint AirWatch Mobile Device Management Integration Module" on page 105
- "SailPoint MobileIron Mobile Device Management Integration Module" on page 109
- "SailPoint Good Technology Mobile Device Management Integration Module" on page 113

> **Note:** **A minority of SailPoint customers have deployed the Integration Modules in this section. SailPoint will provide assistance during the deployment of these integrations. Additional troubleshooting, diagnostic, and best practice information beyond what is contained in this document will be provided on Compass, SailPoint's Online customer portal. In some instances, SailPoint will guide the deployment team and actively participate in the design, configuration, and testing of the integration to the managed system.**
> **For more specific information, refer to the Connector and Integration Deployment Center on Compass.**

Chapter 12: SailPoint AirWatch Mobile Device Management Integration Module

The following topics are discussed in this chapter:

Overview

> **Note:** **SailPoint will provide assistance during the deployment of this Integration Module. Additional troubleshooting, diagnostic, and best practice information beyond what is contained in this document will be provided in the Connector and Integration Deployment Center on Compass.**

This document provides a guide to the AirWatch Mobile Device Management (AirWatch) integration and configuration for your enterprise.

Using this integration, IdentityIQ can retrieve the devices managed by AirWatch, perform operations on them, and manage AirWatch's user account. These entities are managed in IdentityIQ using separate applications named as follows:

- AirWatch Mobile Device Management (MDM) Application (referred to as User Application in this document) for managing AirWatch user accounts
- Device Application (containing the prefix -**Devices**) is created by IdentityIQ during aggregation and is used for managing devices.

Supported features

The AirWatch MDM Integration Module supports the following features:

- Account Management

    - Account Aggregation on the user application to bring in AirWatch user accounts

    - Account Aggregation on the device application to bring in devices managed in AirWatch MDM

    - Delete, Unlock devices

The following table represents what each of the above operation implies to IdentityIQ and AirWatch MDM:

IdentityIQ operation	Resulting change on AirWatch
Account aggregation on user application	The AirWatch user accounts are brought into IdentityIQ.
Account aggregation on device application	The devices that are managed by the AirWatch MDM are retrieved into IdentityIQ.
Delete account for AirWatch device application	Enterprise/Device Wipe and Delete device is triggered on the device via AirWatch MDM system. For more information, see "Operation specific configuration" on page 107.
Unlock account for AirWatch device application	Unlock device is triggered on the device via AirWatch MDM system.
Adding entitlements to account	Adding profiles to device from AirWatch system.
Deleting entitlements from account	Removing profiles from AirWatch system.

- Account - Group Management

  - Device Group Aggregation where device profiles are addressed as groups.

  - Add and remove entitlement (profile) from device.

Supported platforms

SailPoint AirWatch Mobile Device Management Integration Module supports the following version of AirWatch:

- AirWatch Platform Services 8.2

Pre-requisites

Administrator user configured for AirWatch MDM Application must have the following role for provisioning activities:

- REST API Devices Read
- REST API Devices Write
- REST API Devices Execute
- REST API Devices Delete
- REST API Devices Advanced

Note: **IfAirWatch MDM application is behind proxy server, see the "Special Java Considerations" section of the *SailPoint IdentityIQ Installation Guide*.**

Configuration

This section describes the application and additional operation specific configurations.

Application configuration

To create an application in IdentityIQ for AirWatch the following parameters are required:

Parameters	Description
Application Type	AirWatch MIM (Mobile Device Management Integration Module).
AirWatch Server	AirWatch server URL where it's REST API are accessible. For example, https://apidev-as.awmdm.com.
AirWatch Administrator	Administrator of AirWatch server.
AirWatch Administrator Password	Password of the administrator.
API Key	AirWatch server's API key defined for REST API.

Operation specific configuration

This section describes the various configurations required for the following operations:

- Aggregation
- Provisioning

Aggregation

- **Aggregation of devices**: Before aggregating devices against the device application create a correlation rule in the device application to map devices to its AirWatch user. For example, **UserName** is an attribute of the device which specifies the name of the user it belongs to, and **Display Name** of the identity is also **UserName**. So in correlation rule specify application attribute as **UserName** and Identity attribute as **Display Name**.

- **Parameterized device aggregation**: By default AirWatch device aggregation retrieves device profiles and device applications. If you do not want to manage these entities, you can filter them for not being retrieved into IdentityIQ.

  The following configurable parameters impact aggregation:

  - **aggregateDeviceProfile**: (is an application attribute on AirWatch MDM application) determines if the profiles connected to devices are to be retrieved or not. The default behavior is to retrieve the profiles connected to the devices. To change this behavior, set the following value through the debug pages:

    ```
    <entry key="aggregateDeviceProfile” value="false"/>
    ```

  - **aggregateDeviceApp**: (is an application attribute on AirWatch MDM Application) determines if the application installed on devices must be retrieved or not. The default behavior is to retrieve the applications installed on the device. To change this behavior, set the following value through the debug pages:

    ```
    <entry key="aggregateDeviceApp" value="false"/>
    ```

Provisioning

The following provisioning operations are available in IdentityIQ when integrating with AirWatch:

- Delete device

Delete device

- **Delete Device operation from LCM**: In addition to Delete Device, you will be prompted to select **Entire Device Wipe** or **Enterprise Wipe Only** options before deleting the device.

- **Delete Device operation from Certification**: The default wipe operation will be the **Enterprise Wipe Only**. To change this default behavior to **Entire Device Wipe** add the following entry in the application debug of the device application:

  ```
  <entry key="defaultWipeFromCertification" value="Entire Device Wipe"/>
  ```

  Note: If the AirWatch application is already created, update the delete provisioning policy with new field, name as `SecurityPIN` and type as String.

Chapter 13: SailPoint MobileIron Mobile Device Management Integration Module

The following topics are discussed in this chapter:

Overview

> **Note:** **SailPoint will provide assistance during the deployment of this Integration Module. Additional troubleshooting, diagnostic, and best practice information beyond what is contained in this document will be provided in the Connector and Integration Deployment Center on Compass.**

SailPoint IdentityIQ manages MobileIron devices. It does not handle MobileIron users because MobileIron has not provided the supporting User API. It manages these MobileIron entities using the MobileIron REST API's over HTTPS. All the devices are aggregated under MobileIron Mobile Device Management (MDM) Application.

Supported features

The SailPoint MobileIron MDM Connector provides the ability to provision MobileIron devices from IdentityIQ.

The MobileIron MDM Connector supports the following functions:

- Account Management

    - Device aggregation which include device Attribute, Group Attribute - Labels and Entitlements - Apps, Labels.

    - MobileIron MIM support Create, Delete and unlock operations where Create is Registering a Device, Delete is Retire a Device along with Device Wipe that will wipe all the existing device applications and Unlock means unlocking a device by removing a PIN if Set.

- Account - Group Management

    - Device Group Aggregation where groups are Labels in MobileIron.

The following table displays a comparison between operations from IdentityIQ and resultant operations in MobileIron MDM:

IdentityIQ operation	Resulting change on MobileIron
Account aggregation on MobileIron application	The devices that are managed by the MobileIron MDM are retrieved into IdentityIQ with the label attribute represented as the entitlement for that device.
Account group aggregation on MobileIron application	The labels present in the MobileIron MDM are retrieved into IdentityIQ.
Create account of MobileIron application	Registering a device on MobileIron system. For more information, see "Operation specific configuration" on page 111.
Delete account of MobileIron application	Wipe/Retire device is triggered on the device via MobileIron MDM System. For more information, see "Operation specific configuration" on page 111.
Unlock account for MobileIron application	Unlock device is triggered on the device via MobileIron MDM system.
Adding entitlements to account	Adding labels to device from MobileIron MDM system.
Deleting entitlements from account	Removing device labels from MobileIron MDM system.

Supported platforms

SailPoint IdentityIQ MobileIron Mobile Device Management Integration Module supports the following version of MobileIron WebService

- API version for MobileIron WebService 5.7.1

Pre-requisites

Administrator user configured for MobileIron MDM application must have the API role for provisioning activities.

> **Note:** **If MobileIron MDM application is behind proxy server, see the "Special Java Considerations" section of the *SailPoint IdentityIQ Installation Guide*.**

Configuration

This section describes the application and additional operation specific configurations.

Application configuration

To create an application in IdentityIQ for MobileIron the following parameters are required:

Parameters	Description
Application Type	MobileIron MIM (Mobile Device Management Integration Module).

Parameters	Description
MobileIron URL	The URL pointing to the MobileIron system.
MobileIron Administrator	Administrator of MobileIron system.
MobileIron Administrator Password	Password of the administrator.

Operation specific configuration

This section describes the various configurations required for the following operations:

- Aggregation
- Provisioning

Aggregation

MobileIron Device Aggregation retrieves all active status devices, their connected Labels and Applications. IdentityIQ manages labels as entitlement.

Default aggregation without importing or selecting **MobileIron MIM Correlation** for the MobileIron MIM application will create identity in IdentityIQ with **model** name attribute of MobileIron.

For example, `User Name: SM-T301 Account ID: SM-T301`
`        User Name: iPhone4 Account ID: iPhone4`

The **MobileIron MIM Correlation** will display the effect only if identity are already created before account aggregation with same name as of **userDisplayName** attribute of MobileIron.

For example, `User Name: slupinson Account ID: SM-T301,`
`        User Name: AJohn Account ID: iPhone4`

Select the **MobileIron MIM Correlation** under the Correlation tab.

The **MobileIron MIM Correlation** would be populated only if, the following correlation xml input is imported before running the account aggregation task:

```
 <?xml version='1.0' encoding='UTF-8'?>

<!DOCTYPE CorrelationConfig PUBLIC "sailpoint.dtd" "sailpoint.dtd">

<CorrelationConfig name="MobileIron MIM Correlation">

  <AttributeAssignments>

    <Filter operation="EQ" property="displayName" value="userDisplayName"/>

  </AttributeAssignments>

</CorrelationConfig>
```

Provisioning

The following provisioning operations are available in IdentityIQ when integrating with MobileIron:

- Add device
- Delete device

Add device

Default template is present for adding a device (Registering a device to a user). If the user is not present, add a user in the MobileIron system (Local repository) and then assign a device to it.

> **Note:** If new local user is created during "Device Registration" through IdentityIQ then the VSP (MobileIron Server) sets the password for a new local user to the userid.

Valid values for the platform attribute in the **Create Device Provisioning policy** are as follows:

- M - Windows Mobile
- B - BlackBerry
- I - iOS
- S - Symbian
- P - Palm webOS
- A - Android
- L - Mac OS X

Delete device

- **Delete Device via LCM**: In delete device operation, in addition to delete device, user is prompted to select one of the following options:

  - **Device Wipe** (Factory data reset): Device wipe wipes all the data from the device and sets it back to the factory setting.

  - **Enterprise Wipe (Retire)**: Enterprise Wipe removes the MobileIron Client present on the device.

- **Delete Device via Certification**: When a delete device operation is performed from certification, it does not respect the additional operations **Device Wipe** and **Enterprise Wipe**. Hence, IdentityIQ provides an extra MobileIron application attribute named **defaultWipeFromCertification**. Users have to manually add this attribute in the MobileIron MIM application debug page and provide values as **Device Wipe** or **Enterprise Wipe**.

  ```
  <entry key="defaultWipeFromCertification" value="Device Wipe"/>
  ```

  When deleting a device from IdentityIQ using the Device Wipe option, the operation first wipes the device (Data Reset) and then IdentityIQ waits (sleep) for 5 second (default). After the sleep duration IdentityIQ deletes the device from MobileIron system and changes the status from wiped to retire.

  The sleep duration for Wipe operation is a configurable application attribute which can be configured as follows:

  ```
  <entry key="WipeSleepInterval" value="5"/>
  ```

Chapter 14: SailPoint Good Technology Mobile Device Management Integration Module

The following topics are discussed in this chapter:

Overview

> **Note:** **SailPoint will provide assistance during the deployment of this Integration Module. Additional troubleshooting, diagnostic, and best practice information beyond what is contained in this document will be provided in the Connector and Integration Deployment Center on Compass.**

This document provides a guide to the Good Technology Mobile Device Management (MDM) integration and configuration for your enterprise.

Using this integration, IdentityIQ can retrieve the devices managed by Good Technology and perform few operations on them. In addition, this integration also enables IdentityIQ to manage Good Technology's devices. These entities are managed in IdentityIQ using separate applications, named as follows:

- Good Technology MDM Application (referred to as User Application in this document) for managing Good Technology user accounts.
- Device Application (containing the suffix **- Devices**) is created by IdentityIQ during user aggregation and is used for managing devices.

Supported features

The Good Technology MDM Integration Module supports the following features:

- Account Management

    - Account Aggregation on the user application to bring in Good technology role member

    - Device Aggregation on the device application to bring in devices managed in Good Technology MDM

    - Add, Unlock Delete devices

- Account-Group Management:

    - Account Group Aggregation on the user application to bring in Good Technology roles

    - Account Group Aggregation on the device application to bring in policy sets managed in Good Technology MDM

    - Adding/ Removing Entitlement (Policy Set) to device account.

Supported platforms

SailPoint IdentityIQ Good Technology Mobile Device Management Integration Module supports the following version of Good Mobile:

- Good Mobile Control version 2.6.4.972

Pre-requisites

Administrator user configured for Good Technology MDM Application must have the Role with All Rights or following rights for provisioning activities:

- Add handheld for a user
- Add additional handhelds for a user
- Delete handhelds
- Wipe Good for Enterprise app or entire device data
- Reset Good for Enterprise
- Set handheld policy
- View Policy Sets
- Manage policies (handheld and application)
- Manage roles

Configuration

This section describes the application and additional operation specific configurations.

Application configuration

To create an application in IdentityIQ for Good Technology the following parameters are required:

Parameters	Description
Application Type	Good Technology MIM (Mobile Device Management Integration Module).
Good Technology Hostname	Provide computer name of the GMC server.
Good Technology Port	Provide port number of the GMC Web Service. For example, 19005
Good Technology Username	Administrator of GMC server.
Good Technology Password	Password of the administrator.
Page Size	Page size for aggregating devices.

Operation specific configuration

This section describes the various configurations required for the following operations:

- Aggregation
- Provisioning

Aggregation

Aggregation of devices: Good Technology device aggregation retrieves all devices, their connected Policy Set and applications on the device. IdentityIQ manages Policy Set as entitlement.

Default aggregation without creating a correlation rule for the Good Technology Mobile Device Management Integration Module (MIM) application will create identity in IdentityIQ with Display Attribute (Device Model by default) of Good Technology.

For example,

- Name: iPhone 4 (GSM, Rev. A) Account ID: phone 4 (GSM, Rev. A)

   The correlation rule will display the effect only if identity is already created before account aggregation with same name as of correlation attribute of Identity.

- Name: "Demo User1" Account ID: iPhone 4 (GSM, Rev. A)

   Import the following correlation xml which correlates devices to Identity based on name of a user in Good Technology and display name of an Identity:

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE CorrelationConfig PUBLIC "sailpoint.dtd" "sailpoint.dtd">
<CorrelationConfig name="Good Technology Device Correlation">
<AttributeAssignments>
<Filter operation="EQ" property="displayName" value="Name"/>
</AttributeAssignments>
</CorrelationConfig>
```

Provisioning

The following provisioning operations are available in IdentityIQ when integrating with Good Technology:

- Add device
- Unlock device
- Delete device

Add device

While adding a device from IdentityIQ, default provisioning policy accepts the following fields:

- **User DN**: DN of a user present in Good Technology User repository. This is a mandatory field.
- **Messaging Server**: Name of the Good Messaging server. If this field is left blank, IdentityIQ will locate a single GMM server if present. If multiple GMM servers exist, you must provide a value for this field else, the add operation will fail.

Unlock device

While Unlocking device from IdentityIQ, default provisioning policy accepts Unlock Code displayed on the device. Upon successful reset password, temporary unlock code is generated and it will be saved in **Provisioning Request ID** field in Unlock Access Request in IdentityIQ.

Delete device

- **Delete Device operation from LCM**: In addition to delete device, you will be prompted to select **Entire Device** or **Enterprise Data Only** options before deleting the device.
- **Delete Device operation from Certification**: The default wipe operation will be the **Enterprise Data Only**. To change this default behavior to **Entire Device** add the following entry in the application debug of the device application:

```
<entry key="defaultWipeFromCertification" value="Entire Device"/>
```

Assisted Deployment Integration Modules:

IT Security Integration Modules

IT Security Integration Modules (ISIMs) provide enhanced security with improved responsiveness and controls by exchanging identity and access information.

This section contains information on the following:

> **Note:** **A minority of SailPoint customers have deployed the Integration Modules in this section. SailPoint will provide assistance during the deployment of these integrations. Additional troubleshooting, diagnostic, and best practice information beyond what is contained in this document will be provided on Compass, SailPoint's Online customer portal. In some instances, SailPoint will guide the deployment team and actively participate in the design, configuration, and testing of the integration to the managed system.**
> **For more specific information, refer to the Connector and Integration Deployment Center on Compass.**

SailPoint Integration Guide

Chapter 15: SailPoint HP ArcSight Integration Module

The following topics are discussed in this chapter:

Overview

Note: **SailPoint will provide assistance during the deployment of this Integration Module. Additional troubleshooting, diagnostic, and best practice information beyond what is contained in this document will be provided in the Connector and Integration Deployment Center on Compass.**

HP ArcSight ESM is a Universal log management solution that helps enterprises identify and prioritize current and potential security threats. SailPoint IdentityIQ collects the security event information such as Audit information. The SailPoint IdentityIQ integration with HP ArcSight allows both end systems to take remediation action in case of security threats.

IdentityIQ integration with ArcSight enables the following scenarios:

1. IdentityIQ data (Identity, Account, Audit, and Syslog) stored in IdentityIQ can be exported to ArcSight. ArcSight administrator can store this data in an ArcSight Active List.
    IdentityIQ data can be exported to ArcSight for correlation, such as successful provisioning of privileged accounts, password changes, login failure and so on. For more information on ways to export data, see "Export from IdentityIQ to ArcSight" point in "Supported features" section.

2. IdentityIQ can import filtered activity event data from ArcSight; based on which activity-based remediation processing can be triggered. Event records are expected in standard ArcSight Common Event Format (CEF). Events received are matched with users held within the IdentityIQ warehouse, and used to trigger activity policies when certain types of event are recognized. These triggers result in a business process being executed which generates a full re-certification for the affected user, and also causes a re-calculation of the user's risk score and update of risk reports and dashboard content to highlight the activity.

Note: **Creating an ArcSight Active Channel or Active List is outside the scope of this document. This document assumes the ArcSight administrator is familiar with steps to create an ArcSight Active Channel or Active List. It provides the IdentityIQ information an ArcSight administrator will require to create an ArcSight Active Channel or Active List.**

Common Event Format (CEF)

CEF is an extensible, text-based, high-performance format designed to support multiple device types in the simplest manner possible. CEF defines syntax for log records comprised of a standard header and a variable extension, formatted as key-value pairs. CEF uses syslog as a transport mechanism and the following format, comprised of a syslog prefix, a header and an extension, as shown below:

For example,

```
Jan 18 11:07:53 host CEF:Version|Device Vendor|Device Product|Device
Version|Signature ID|Name|Severity|[Extension]

Dec 19 08:31:10 host CEF:0|Security|threatmanager|1.0|101|out of hours workstation
login|10|suser=hbutler src=activedirectorydomain ip=10.1.76.224
```

Supported features

- **Export from IdentityIQ to ArcSight**: The IdentityIQ data can be exported in:
  - **Flat file in CEF**: Using Advanced Analytics we can export Identity, Account, Audit and Syslog data in CEF.
  - **Database tables**: The **ArcSight Data Export** task enables you to export Identity (which includes account and identity data) and Audit data to external tables.
- **Import into IdentityIQ from ArcSight**: This integration supports including event logging data from ArcSight and associate it to Identities in IdentityIQ so that potential policy violations can be triggered or provide greater visibility as part of access reviews as to any suspicious or error prone access a user may have.

Pre-requisites

(*Applicable for import of ArcSight Events into SailPoint IdentityIQ*) At least one application must be configured in SailPoint IdentityIQ and Users/Groups aggregated into SailPoint IdentityIQ system.

> **Note:** **Users present in HP ArcSight must also be present in SailPoint IdentityIQ.**

Configuration

This section describes the general, operation specific configurations and the steps that must be performed to configure the HP ArcSight Integration Module

Configuration to export IdentityIQ Data to ArcSight

The Identity, Account and Audit information from SailPoint IdentityIQ can be exported to ArcSight using CEF flat file or database:

1. Export data from SailPoint IdentityIQ to ArcSight tables.
2. Export data from SailPoint IdentityIQ to Flat file in Common Event Format.

Export Data from SailPoint IdentityIQ to ArcSight tables

The **ArcSight Data Export** task enables you to export Identity and Audit data to external tables. You can select to export Identity information and Audit events from IdentityIQ Database.

Create the export databases on your destination data source before using the ArcSight Data Export task.

1. Navigate to **Monitor => Tasks**.

2. Create a new ArcSight Data export task.

3. Provide the Data Source Parameters.
   ArcSight Data Export options are:

Options	Description
Datasource Parameters	
Database	Select a database type from the drop-down list.
User Name	Enter the user name parameter of the database.
Password	Enter the password of the database.
Driver Class	Enter the driver class used for the database.
URL	Enter the URL of the database.

4. Click on **Generate table Creation SQL** to generate table's schema and create database that includes export tables which you can hand off to a database administrator for execution.
   The task adds the following tables in database:

Tables	Description
sptr_arcsight_export	Table to maintain the task execution history.
sptr_arcsight_identity	Table contains exported data of Identity.
sptr_arcsight_audit_event	Table contains Audit Events information.

5. Select **Object Export** options.
   The **Object Export** options are:

Options	Description
Export Identities	Select the check box to export Identity related data in ArcSight tables. It provides the following options: • **Full**: Exports all the records irrespective if they were exported earlier. • **Incremental**: Exports only records that are updated since last run of this task. This option can even be selected when running the task for first time. When the task is running for first time, this option exports all records similar to the **Full** option.
Export Audits	Select the check box to export Audit Events in ArcSight table. It provides the following options: • **Full**: Exports all the records irrespective if they were exported earlier. • **Incremental**: Exports only records that are updated since last run of this task. This option can even be selected when running the task for first time. When the task is running for first time, this option exports all records similar to the **Full** option.

6. After completing the customizing report options, click **Save** for later use or Save and Execute to save the report and run it immediately.

Configuring HP ArcSight Task to populate host name or IP

The value of column application_host can be populated by adding a map **arcsightAppNameHostMap**. Adding the **arcsightAppNameHostMap** map the administrator configuring this integration can define the hostname (or IP address) which must be used for an Account. It is recommended this hostname (or IP address) is same as the configured in the ArcSight configuration.

The **arcsightAppNameHostMap** map must be defined in the **ArcSight Data Export** Task created above. The key in the map should be name of the application defined in IdentityIQ and value should be hostname, IP, or any string that ArcSight administrator understands.

1. To add the map, navigate to debug page, navigate to TaskDefinition and open the ArcSight task configured above.

2. Add the entry as key = Name of Application defined in IdentityIQ and value as the string to identify host of Account like Hostname or IP.

3. Save the task definition.
   For example:

```
<entry key="arcsightAppNameHostMap">
    <value>
        <Map>
          <entry key="LinuxApp1" value="linux01.sailpoint.com"/>
          <entry key="LinuxApp2" value="127.15.19.21"/>
          <entry key="ADDirectApp" value="AD.sailpoint.com"/>
          <entry key="ServiceNowApp" value="https://sailpoint.service-now.com"/>
          <entry key="ACF2App" value="ACF2-Mainframe"/>
        </Map>
    </value>
</entry>
```

Note: **If the application name is not defined in the map the host field will be blank.**
As mentioned above, this document provides the information an ArcSight administrator requires to create an ArcSight Active List or Active Channel. The information below provides the same. Following fields are added in export table:

Table 1—IdentityIQ sptr_arcsight_identity export table

Fields	Description
linkid	Primary key for Link table in IdentityIQ database. This field will be copied from spt_link table id field. This will be the primary key for export table.
identityid	Primary key in Identity table. This field will be copied from spt_Identity table.
modified_dt	Populates timestamp when the record will be exported in export table. The field can be referred while configuring time based ArcSight database connector.
identity_display_name	Represents Display Name of Identity which will be copied from spt_identity table field (display_name).

Table 1—IdentityIQ sptr_arcsight_identity export table

Fields	Description
identity_firstname	Represents first name of Identity which will be copied from spt_identity table field (firstname).
identity_lastname	Represents last name of Identity which will be copied from spt_identity table field (lastname).
application_type	Populates the type of Account which is connected to the Identity like ActiveDirectory – Direct, ACF2 – Full, Box, Cloud Gateway, ServiceNow and so on.
application_host	The host name, IP, or any string which can be used by ArcSight administrator to identify the host of link/account uniquely. Customer can enter any string which can be sent to ArcSight to identify the host of link. This field can be populated as explained in "Configuring HP ArcSight Task to populate host name or IP" on page 122.
application_name	Populates the name of Application of the Account connected to the Identity.
link_display_name	The account connected to the identity which will be copied from spt_link table, field display_name.
entitlements	Represents comma separated list of entitlements to the link of Identity.
risk_score	Represents the composite risk score of Identity.

Table 2—IdentityIQ sptr_arcsight_audit_event export table

Fields	Description
auditid	The audit ID which is primary key for the export Audit table. The field will be copied from spt_audit_event table id field.
created_dt	Populates timestamp when the record will be exported in export table. The field can be referred while configuring time based ArcSight database connector.
owner	Describes the Owner of the audit generated.
source	Provides more details to help ArcSight administrator determine the source of audit.
action	Describes the action taken on entity.
target	Provides target details.
application	Describes the name of application the target belongs to.
account_name	The name of Account is populated in this field.
attribute_name	The name of attribute modified.
attribute_value	The value provided to the attribute.

Export Data from SailPoint IdentityIQ to Flat file

1. Navigate to **Analyze => Advanced Analytics**.

2. Navigate to Identity Search/ Audit Search/Account Search Tab.

3. Select the Search Criteria and Fields to display.

4.   Click on **Run Search**.

5.   Click on **CEF Flat file** export button to export search results to file in CEF.
     The Search Results page have the following options to save:

     -   **Save Search**: It is used to save the search criteria and fields to display.

     -   **Save Search as Report**: These type of reports can be accessed as a report, as schedules or for execution by performing the procedure:
         a.   Navigate to **Analyze => Reports**.
         b.   Right click on the report and schedule or execute the report.
         c.   Navigate to Report Results tab to see the report result.
         d.   Click on the **Report**.
         e.   Click on the **CEF Flat file** export button to export the report to file in CEF.

             This will generate a file with data in CEF which can be used by ArcSight to import events in ArcSight ESM.

     Note: For more information on Advanced Analytics, see *SailPoint IdentityIQ User Guide*.

Configuration to Import HP ArcSight CEF Flat File to SailPoint IdentityIQ

1.   Access the Application Configuration Console.

2.   Navigate to Schema tab.
     Mark the field as correlation key for which you want to correlate activity from HP ArcSight to SailPoint IdentityIQ.

     For Example: `sAMAccountName` for Active Directory application.

3.   Navigate to Activity Data Sources tab.

     Note: For more information on Activity Data Source, see *SailPoint IdentityIQ Administration Guide*.

4.   Click on **Add** to add new Activity Data Source.

5.   Select Activity Data source Type as **CEF Log File**. The default Transformation rule and Correlation rule will be automatically selected.

     Note: You can change the value of `cefLinkAttribute` in correlation rule to set correlation key as per application.

6.   Navigate to Transport Settings tab, select the Transport Type as local, ftp or scp.

7.   Navigate to Log File Settings tab and in the File name provide the exact path of the CEF Flat file. For example, `C:\ArcSight\activedirectory.csv`

8.   Click on **Save** button to save activity data source configuration.

9.   Click on **Save** button to save the application.

If the correlation key is not marked and aggregation of account for that application is already performed, then perform the following:

- Access the Application Configuration console.

- Navigate to the Correlation tab.

- Click on **New** button to create a new Account Correlation.

- Click on next button and provide the name of the configuration.

- Select the Application Attributes and Identity Attributes and click on **Add** button.

- Click on **Save**.

- Click on **Save** to save the application.

  After the correlation configuration is done, execute the account aggregation (with optimization turned off to pick up the existing accounts) again.

10. Navigate to **Define => Identities**.

11. Click on the identity for which you want to enable Activity monitoring and import data from ArcSight.

12. Navigate to Activity Tab.

13. Select the **Activity Monitoring** checkbox.

14. Save the Identity.

15. Navigate to **Monitor => Tasks**.

16. Create a new Activity Aggregation Task.

17. Select an activity data source which is configured above in Step 8.

18. Save and execute the task.

    - To see the result of the task executed in previous step navigate to Task Results tab and click on the task.

    - To see the correlated events navigate to **Define => Identities**. Select the identity for which you have correlated the event. Navigate to Activity Tab. Check the Recent Activities section.

    Note: **After correlating the HP ArcSight event to Identity, the Policy Violation and Certification can be created and used to notify for any activity for that identity using the workflow.**

Chapter 16: SailPoint Lieberman Integration

The following topics are discussed in this chapter:

Overview

> **Note:** **SailPoint will provide assistance during the deployment of this Integration Module. Additional troubleshooting, diagnostic, and best practice information beyond what is contained in this document will be provided in the Connector and Integration Deployment Center on Compass.**

IdentityIQ Lieberman Integration can be used to manage Privileged Users in various target systems. Lieberman Enterprise Random Password Manager (ERPM) is used to manage password of privileged accounts on various target systems. Using this integration IdentityIQ Certifiers can certify the access and permissions assigned to the privileged users.

The IdentityIQ Lieberman Integration is built upon IdentityIQ Active Directory application which must be configured to use this integration. After configuring IdentityIQ Active Directory, Lieberman Target Collector must be configured on the same. The Lieberman Target Collector aggregates permissions assigned to Domain Users and Domain Groups (Active Directory User/Groups) on different management sets in Lieberman ERPM. The aggregated permissions are associated with the Accounts and Groups associated with the Active Directory application.

This document assumes IdentityIQ Active Directory application has been configured, and lists the steps to configure Lieberman Target Collector. For more information on configuring IdentityIQ Active Directory application, see *SailPoint Direct Connectors Administrator and Configuration Guide*.

Supported platforms

SailPoint Lieberman Integration Module supports the following version of Lieberman:

- Lieberman ERPM 4.83.9 SR1

Configuration

Lieberman target collector requires **msDS-PrincipalName** field of Active Directory account and group to be marked as Correlation Key.

The unstructured target defined on **Unstructured Targets** tab of application would be used by the Target Aggregation Task to correlate targets with permissions assigned to Identities and Account Groups.

These permissions can be revoked as part of certification remediation process, the certifier of the Identity or Entitlement can revoke these permission.

The Lieberman Target Collector configuration parameters are as follows:

Field	Description
Authenticator	The Authenticator of the user. The different Authenticators supported by IdentityIQ are: • **Explicit User***:* The user is Lieberman user and will be authenticated by Lieberman. • **Domain User***:* The user is Domain user and will be authenticated by the Active Directory.
URL*	URL to the web service.
Username*	Username of the Lieberman Administrator. It should be prefixed with domain name as (Domain\Username) in case of Domain User authenticator.
Password*	Password of the Lieberman Administrator.
Rules: Specify the rule used to correlate the targets with appropriate AD accounts. **Note: Click the "…" icon to launch the Rule Editor to make changes to your rules if needed.**	
Correlation Rule	Used to determine how to correlate target permission information to Active Directory users in IdentityIQ. Default correlation rule for Lieberman can be found in **targetRuleLibraries.xml** with **Lieberman Default Target Correlation For Active Directory Application** name**.**

Troubleshooting

1 - Target Aggregation displays the time out exception error for larger data.

Target Aggregation displays the following timeout exception error for larger data:

```
Target Source Scan failed. Reason: openconnector.ConnectorException: Exception
occurred while processing permissions, check the configuration: Read timed out
```

Resolution: Add the following in Target Source on debug page and set the timeout value.

```
<entry key="clientOptions">
  <value>
    <Map>
      <entry key="CONNECTION_TIMEOUT">
        <value>
          <Integer>180000</Integer>
        </value>
      </entry>
      <entry key="SO_TIMEOUT">
        <value>
          <Integer>180000</Integer>
        </value>
      </entry>
    </Map>
  </value>
</entry>
```

Note: Above Time Out value can be change as per requirement(s).

2 - While performing target aggregation from Lieberman ERPM, an error message was displayed.

The following exception occurred while processing permissions, check the configuration:

```
org.apache.axis2.databinding.ADBException: Unexpected subelement
{http://schemas.datacontract.org/2004/07/RouletteAppService_WCF}PermissionRequestRe
moteAccess
```

Resolution: User must generate the new stub classes form WSDL.

Perform the following steps to generate the stub classes:

1. Open the following URL:
   http://*hostMachineName*/ERPMWebService/AuthService.svc

   where *hostMachineName* is your host computer name.

   On the web page that is displayed, click on the first link. For example,
   http://*hostMachineName*/ERPMWebService/AuthService.svc?wsdl

   The WSDL page is displayed. Right click on the page and save the page by selecting the **Save page as** option.

2. Axis2 is used to generate stub classes and can be downloaded from apache Axis2 version 1.6.1
   http://axis.apache.org/axis2/java/core/download.cgi

3. Extract downloaded **Axis2** file and set the **AXIS2_HOME** environment variable pointing to the **Axis2** directory.

4. Set the **JAVA_HOME** environment variable pointing to the **jdk** directory.

5. Navigate to **%AXIS2_HOME%\bin** directory and run the following command:
   ```
   wsdl2java.bat -uri <path>\AuthService.svc.xml -p sailpoint.unstructured.lieberman
   -o <path>
   ```
   This creates the java files in given path with the given package name as follows:
   ```
   sailpoint.unstructured.lieberman
   ```

6. To compile these classes, add the following jar files from **%AXIS2_HOME%\lib** into the classpath along with the current directory.
   ```
   set
   classpath=.;%AXIS2_HOME%\lib\axiom-api-1.2.12.jar;%AXIS2_HOME%\lib\axiom-impl-1.
   2.12.jar;%AXIS2_HOME%\lib\axis2-adb-1.6.1.jar;%AXIS2_HOME%\lib\axis2-kernel-1.6.
   1.jar;%AXIS2_HOME%\lib\axis2-transport-http-1.6.1.jar;%AXIS2_HOME%\lib\axis2-tra
   nsport-local-1.6.1.jar;%AXIS2_HOME%\lib\geronimo-stax-api_1.0_spec-1.0.1.jar;%AX
   IS2_HOME%\lib\neethi-3.0.1.jar;%AXIS2_HOME%\lib\wsdl4j-1.6.2.jar;%AXIS2_HOME%\li
   b\wstx-asl-3.2.9.jar;%AXIS2_HOME%\lib\XmlSchema-1.4.7.jar;%AXIS2_HOME%\lib
   ```

7. Compile the classes using the following command:
   ```
   %JAVA_HOME%\bin>javac <path>\src\sailpoint\unstructured\lieberman\*.java
   ```

8. Copy the newly generated stub classes from ***path*\src\sailpoint\unstructured\lieberman** directory into **\identityIQ\WEB-INF\classes\sailpoint\unstructured\lieberman** directory.

9. Restart IdentityIQ.

3 - Mismatch between the WSDL and stub classes.

Generate the stub classes by performing the following steps in case there is a mismatch between the WSDL and stub classes:

Troubleshooting

1. Use the following command to generate stub java classes at `<output-directory>/src/sail-point/unstructured/Lieberman` directory:
   `wsdl2java.bat -uri Lieberman//wsdl/AuthService.svc.xml -p sailpoint.unstructured.lieberman -o <output-directory>`

2. Compile these classes using the `build.xml` file generated in <output-directory> using the ant command.

   Note: Ensure that Apache-Ant is installed before running the ant command.

3. Copy the generated classes in `D:\GitHub\identityiq\build\WEB-INF\classes\sail-point\unstructured\lieberman` directory.

Assisted Deployment Integration Modules:
SAP GRC Integration Module

The SailPoint SAP GRC Integration Module uses the SAP GRC Access Control web services which are provided by SAP for Access Request Management (ARM).

This section contains information on the following:

- "SailPoint SAP GRC Integration Module" on page 133

Note:	A minority of SailPoint customers have deployed the Integration Module in this section. SailPoint will provide assistance during the deployment of this integration. Additional troubleshooting, diagnostic, and best practice information beyond what is contained in this document will be provided on Compass, SailPoint's Online customer portal. In some instances, SailPoint will guide the deployment team and actively participate in the design, configuration, and testing of the integration to the managed system.
	For more specific information, refer to the Connector and Integration Deployment Center on Compass.

SailPoint Integration Guide

Chapter 17: SailPoint SAP GRC Integration Module

The following topics are discussed in this chapter:

Introduction

This chapter provides a guide to the integration between SAP GRC (Governance, Risk and Compliance) and IdentityIQ. This integration is used to leverage SAP GRC's ability to perform SOD (Separation of Duties) checks and take remediation or mitigation decisions within the SAP GRC. The mitigation decision must be taken in SAP GRC so that SAP GRC is aware of the mitigation controls which is applied on risks and would not report these risks till the time mitigation is applicable.

SailPoint SAP GRC Integration Module uses the SAP GRC Access Risk Analysis (ARA) and Access Request Management (ARM) web services which must be enabled before using the integration.
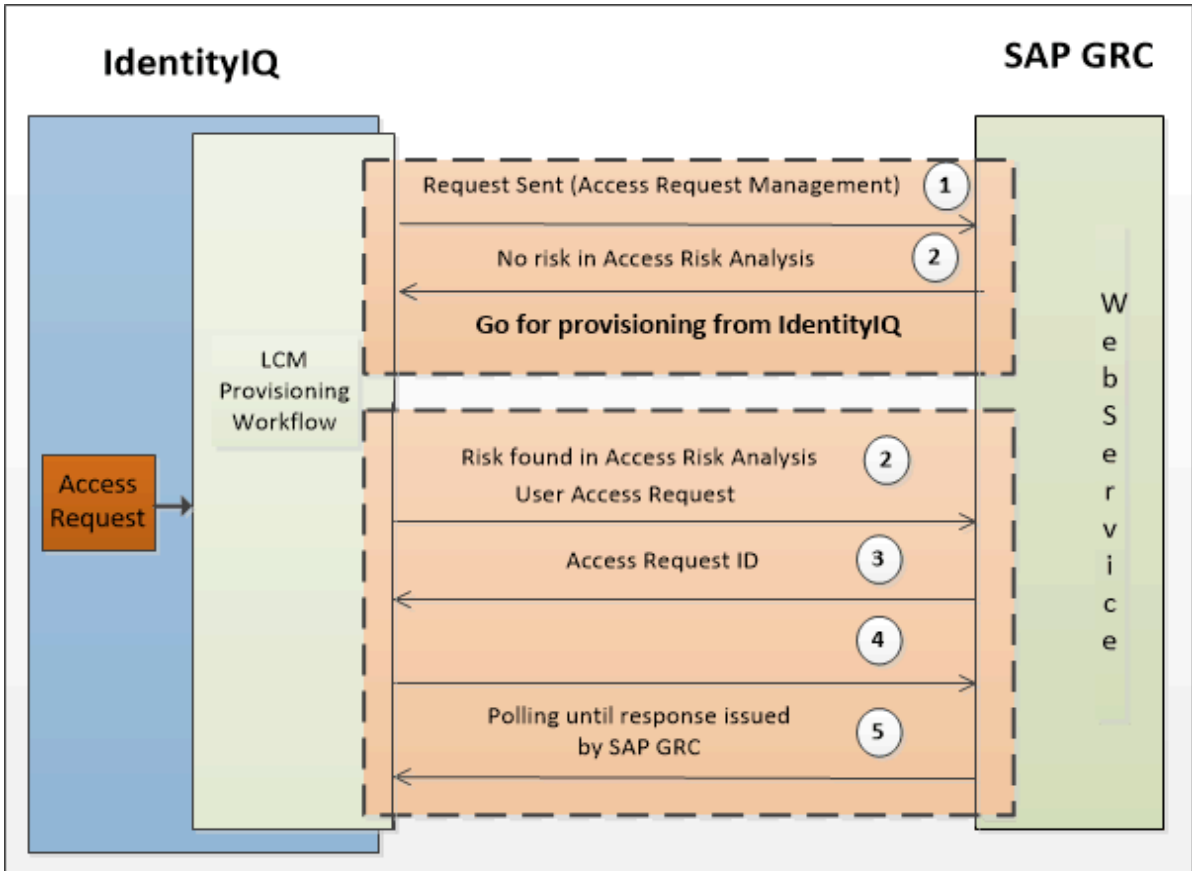
Figure 1—SAP GRC Integration Module with IdentityIQ

SailPoint SAP GRC Integration Module (Figure 1) enables checking for risk in the request placed in IdentityIQ (containing SAP Direct Roles and Profiles) in the following method:

1. Request will be sent to the SAP GRC for proactive check.

2. ARA Web Service will check for the risk present in the request, if no risk is returned then IdentityIQ will con-tinue provisioning the request.

3. If ARA Web Service returns the risk in the request, then corresponding request is created in SAP GRC using the ARM Web Service.

4. IdentityIQ will continue with polling the request until response issued by SAP GRC.

5. On the basis of the response returned in step 4 above (approval or rejection by SAP GRC), IdentityIQ will continue with provisioning or rejection of the request.

Supported features

The SAP GRC Integration Module performs Risk Analysis for new and change account requests using Lifecycle Compliance Manager (LCM).

Note: **SAP GRC Integration supports Basic Authentication level with Transport Channel Authorization as User ID/Password.**

Supported platforms

SailPoint SAP GRC Integration Module supports the following version of SAP GRC Access Control:

- SAP GRC Access Control 10.1

Pre-requisites

- SAP GRC Server Settings
- SAP Connector changes for supporting SAP GRC integration
- Creating IdentityIQ application of type SAP GRC
- SAP GRC workflows

SAP GRC Server Settings

- Perform the following settings on the SAP GRC Server using the administrator privileges:
    a. Execute **SPRO** transaction code and click **SAP Reference IMG** button.
    b. Expand **Governance**, **Risk and Compliance** ==> **Access Control** ==> **User Provisioning** option.

    c. Click on image to execute **Maintain Provisioning Settings** transaction.

    d. In Dialog Structure double click on **Maintain Global Provisioning**.
    e. In Provisioning options section, select **No Provisioning** from the drop down against label **Auto Provisioning**. The default value for this label is **Auto provisioning at end of request**.
- A SAP ABAP type of connection must be defined in SM59 transaction, which would be used to indicate IdentityIQ connection virtually at SAP GRC server. This connection would be treated as Request Initiation System in SAP GRC application configuration. For more information, see Creating IdentityIQ application of type SAP GRC .
- Status of requested Roles must be set to production on the SAP GRC Server.

Perform the following steps to obtain the SAP GRC URLs required when configuring the SAP GRC application in IdentityIQ

Note: **Following steps are provided, considering that all required web services are set to active mode.**

1. Execute **SOAMANAGER** transaction code on SAP GRC server.

2. Under Service Administration, select **Web Service Configuration**.

3. In Search criteria, select **Object Type** as Service Definition and Object Name contains **GRAC***.

4. Click on **Search**.

5. In search result, search (User Access web service) and click `GRAC_USER_ACCESS_WS` web service.

6. Perform the following for `GRAC_USER_ACCESS_WS` web service:

   - Click on icon to open Service WSDL Generation.

   - Copy the URL from WSDL Generation section and open this URL in browser.

   - Locate the following string in the XML and copy the binding URL (in bold) mentioned in the string:

     ```
     <wsdl:port name="Web_Service_BINDING_soap12"
     binding="tns:Web_Service_WS_BINDING_soap12">

     <wsoap12:address location="http://XXXX"/>

     </wsdl:port>
     ```

7. Perform the above steps for the following respective web services:

   - Risk Analysis: GRAC_RISK_ANALYSIS_WOUT_NO_WS

   - Request Details: GRAC_REQUEST_DETAILS_WS

   - Audit Log: GRAC_AUDIT_LOGS_WS

   These URLs would be used for User Request, Request Details, Risk Analysis, and Audit Log respectively in SAP GRC application configuration in IdentityIQ. For more information, see "Creating IdentityIQ application of type SAP GRC " on page 136.

SAP Connector changes for supporting SAP GRC integration

For supporting the SAP GRC integration, create a SAP GRC RFC (ABAP) connection on the SAP GRC system. For more information on creating the connector on SAP GRC system, see "Creating a RFC Connection on SAP GRC system" on page 145.

On SAP Direct application configuration page, the following checkbox and field have been introduced:

- **Enable SAP GRC**: Select this checkbox for the SAP GRC application to be sent to SAP GRC server for risk analysis.
- **SAP GRC Connector Name**: The value of this field would be the name of the SAP GRC Connector created in "Creating a RFC Connection on SAP GRC system" section.

In case of single SAP Direct application, user can add connector name manually. For multiple SAP Direct applications, a rule is provided to avoid the manual work. For more information, see "Importing SAP GRC Application Rule" on page 144.

Creating IdentityIQ application of type SAP GRC

Note: **When you create your SAP GRC application you can configure it and perform a test connection from the Application Definition page; however, once you have saved the SAP GRC application, you must use the Debug page to view or edit it. It will not appear in your list of applications in the Application Definition page.**

IdentityIQ application holds the connection parameters to communicate with SAP GRC server. Following are the connection parameters required by SAP GRC Server:

Field names	Description
SAP GRC Connection Settings	
Username*	User Name from SAP GRC Server which have minimum permissions.
	For more information on minimum permissions required by the SAP GRC user, see "Minimum permissions required for SAP GRC user" on page 140.
Password*	Login Password.
Request Initiation System *	Name of the connector configured in SAP GRC server which is treated as **Request Initiation System**. This connector is configured in SPRO define connectors or in SM59 transaction code.
	For more information, see "Creating a RFC Connection on SAP GRC system" on page 145.
Polling interval	Polling interval in minutes (Range 1 to 360).
Web Service URL Details	
User Access *	End Point URL for SAP GRC User Access Web Service.
	Format of URL would be as follows:
	http://<SAP GRC Host Name>/sap/bc/srt/rfc/sap/grac_user_acces_ws<WebService Binding URL>
Risk Analysis*	End Point URL for SAP GRC Access Risk Analysis Web Service.
	Format of URL would be as follows:
	http://<SAP GRC Host Name>/sap/bc/srt/rfc/sap/grac_risk_analysis_wout_no_ws<WebService Binding URL>
Request Details *	End Point URL for SAP GRC Request Detail Web Service.
	Format of URL would be as follows:
	http://<SAP GRC Host Name>/sap/bc/srt/rfc/sap/grac_request_details_ws<WebService Binding URL>
Audit Log	End Point URL for SAP GRC Audit Log Web Service.
	To get Mitigation details in IdentityIQ, audit log URL can be provided. This detail can be viewed in Interaction section of Access Request Page.
	Format of URL would be as follows:
	http://<SAP GRC Host Name>/sap/bc/srt/rfc/sap/grac_audit_logs_ws<WebService Binding URL>

Field names	Description
Note: For more information, see "Perform the following steps to obtain the SAP GRC URLs required when configuring the SAP GRC application in IdentityIQ" on page 135.	

SAP GRC workflows

The standard LCM provisioning workflow does not support the SAP GRC integration. Custom workflows are shipped with IdentityIQ to support this integration.

Integration workflows

Following are the custom workflows to interact with SAP GRC:

- SAP GRC Data Generator

    - Gathers all provisioning request from IdentityIQ.

    - Filter the plans which contain roles from SAP Direct application which has SAP GRC check box enabled.

        For more information, see "Custom workflows provided for SAP GRC integration" on page 141.

    - Creates a map of all the requested items which are required by SAP GRC Request Executor.
      Note: The step to create map from the plan can be customized as required.

- SAP GRC Request Executor

    For a proactive check performed on Access Request, if there is no risk found for particular Access Request then request will be provisioned, else perform the following:

    a. Creates a request on SAP GRC Server.
    b. Polling is done for the request till it is in pending status.
    c. Receives the response back from SAP GRC Server.
    d. Based on the response, this workflow takes decision whether to provision the request on SAP Server or not.

        For more information, see "Custom workflows provided for SAP GRC integration" on page 141.

    Note: Proactive check on Access request displays the risks even if they are mitigated earlier. Therefore each time mitigated risks get calculated, request would be created on SAP GRC for approval.

Importing integration workflows

Import `Workflow_SAPGRC_Integration.xml` which contains **SAP GRC Data Generator** and **SAP GRC Request Executor** workflows located at `../WEB-INF/config` file.

These workflow must be integrated in LCM provisioning workflow in **Provisioning Approval Subprocess** sub-process as mentioned below:

1. Change **Provisioning Approval Subprocess** as mentioned below:

   - Navigate to process designer and click on **Add A Step**.

   - Select **Stop.**

   - Drag and drop the **Stop** step (in Auto Layout) after the **end** step.

   - Right click on **end** step and select **Change Icon**.

   - Select **Generic** and click on Save.

   - Right click on **end** and click **Edit Step**.

     Provide the following values in the Details section:

     - **Name:** Invoke SAP GRC Data Generator

     - **Subprocess:** (select under Action section) SAP GRC Data Generator.

       Save the form.

   - Right click on **Stop** step, click **Edit Step** and in Details section provide the name as **end**.

     Save and close the workflow.

   - Right click on **Invoke SAP GRC Data Generator** step and perform the following:
     a. Start the transition and end that transition on **end** step.
     b. Save the changes.
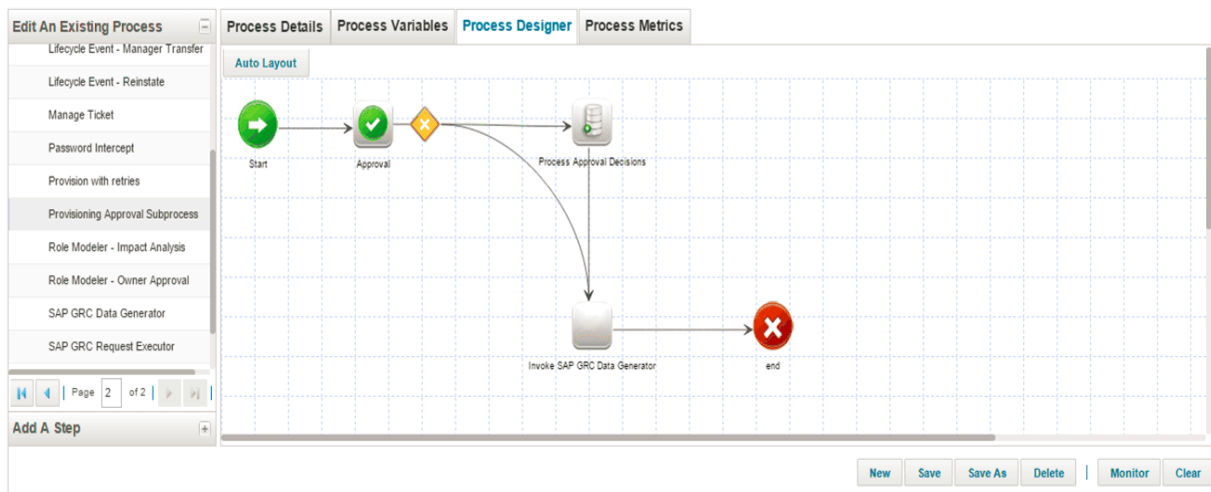
   - Open the **Provisioning Approval Subprocess** and right click on **Invoke SAP GRC Data Generator** and edit the step.

     In Arguments section of this step search for identityName, identityDisplayName, project, approvalSet and enter the values as identityName, identityDisplayName, project and approvalSet respectively for Reference fields. Save the changes.

   - Save the changes.

- Navigate to debug page and search the following in **Provisioning Approval Subprocess** workflow:

  ```
  <Step icon="Default" name="Invoke SAP GRC Data Generator"
  ```

  Perform the following change:

  ```
  <Step icon="Default" name="Invoke SAP GRC Data Generator" posX="320" posY="196"
  resultVariable="approvalSet">
  ```

  After all the <Arg> tags add the following before invoking the SAP GRC Data Generator workflow:
  ```
  <Return name="approvalSet" to="approvalSet"/>
  ```

2. Open **SAP GRC Data Generator** process and perform the following:
   In Process Variable section open **applicationNameSAPGRC** variable and in Initial value section select String and provide value as the name of application of type SAP GRC configured in IdentityIQ.

(*Optional*) Life cycle event workflows

1. Import `Workflow_SAPGRC_LifeCycle_Events.xml` located at `../WEB-INF/config` which includes **New Account - Joiner** and **Mover - Process** workflows.

2. These Lifecycle events are triggered in case of joiner and change attribute events respectively.

   Note: **These are sample workflows which can be customized as required.**

Minimum permissions required for SAP GRC user

The minimum permissions required for the SAP GRC account must have the following Authorization Objects assigned to it:

Authorization Objects	Field names
S_Develop	- Activity: Create or generate, change, display, Delete, activate, generate - Package: * - Object name: * - Object Type: * - Authorization group ABAP/4 pro: *
GRAC_SYS	- Activity: Create or generate - Application Type: 001 (SAP) - SYSTEM Environment: Development, Production, Test - Connector ID: * or select the connector id configured on SAP GRC
GRAC_REQ	- Activity: Create or generate - Business Process: BS00,FI00 - Functional Area: * or provide the required functional areas - Request for single or multiple: All values - Request Information: All values - Request Type: 001,002

Authorization Objects	Field names
GRAC_ROLED	• Activity: All values • Business Process: * • Connector Group: * or select the connector group configured on SAP GRC • Role Sensitivity: * • Role Type: BUS, COM, CUA, PRF, SIN • Role Name: *
S_SERVICE	• Program, transaction or function: * • Types of Check Flag and Authorization: All Values
GRAC_RA	• Activity: Administrator • Object types for Authorization: User, Role, Profile • Risk Analysis Mode: All values • Report Type: All values
GRFN_CONN	• Activity: Execute • Connector ID: *

Custom workflows provided for SAP GRC integration

SAP GRC Data Generator

This workflow fetches following information from IdentityIQ:
- Details of user for whom access is requested.
- Details of user who is requesting for access.
- Details of access which are requested.

SAP GRC Data Generator subprocess has a custom script to fetch the values which can be changed by user as per his requirements.

Inputs to SAP GRC Data Generator workflow

Field names	Description
identityName	Name of the identity object being modified.
plan	A master provisioning plan object required for building transient approval set for SAP GRC response.
project	The provisioning plan ready to execute.
identityDisplayName	Display name for identity.

Field names	Description
applicationNameSAPGRC	A **ProvisioningProject** object describing the modifications to the identity. This may include a list of Question objects which will cause the generation of a Form and a WorkItem to solicit additional information necessary for provisioning.
approvalSet	This attributes is set during the **Build Approval Set** step, which builds this list by going through the ProvisioningPlan to build the line items that must be approved. This variable includes all ApprovalItems that are part of the request process and is updated during the AfterScript of the approval process by assimilating the decisions and comments from the Approvals copy of the ApprovalItem.
requester	Requester who initiated the request.
trace	Used for debugging this workflow and when set to true, trace will be sent to **stdout**.

Output of SAP GRC Data Generator workflow

Table 1— Output of SAP GRC Data Generator workflow

Field names	Description
completeDetailMap	Map used to keep all other maps required by SAP GRC Request Executor.
userInfoMap	Map used to provide details for the link to whom access request is created.
requestedLineItemMap	A map with details of roles which are requested for a link.
credentialsMap	Map containing values of credential to connect to SAP GRC server.
requestHeaderDataMap	Map containing values of requester.
userGroupsMap	Map containing user group details.
accountRequestSAPGRC	A list of SAP Direct AccountRequest which are qualified for SAP GRC violation check.
customFieldsValMap	Map containing custom values of requester.
parameterMap	Map containing parameter values to be set.
language	Language used by requester. Default: English.

The **Invoke SAP GRC Request Executor** step of SAP GRC Data Generator workflow invokes the SAP GRC Request Executor workflow.

SAP GRC Request Executor

The SAP GRC Request Executor workflow pro-actively checks for Access Request Risk with SAP GRC, if risk is found then it creates the request on SAP GRC server and regularly checks the status of the request in asynchronous manner. As this workflow checks status of the response at regular interval, variables related to such polling are defined here. User can change those variables as per requirement.

Inputs to SAP GRC Request Executor workflow

Field names	Description
numberOfRetries	The number of retries that will be attempted before failure of the provisioning activities.
retriableErrors	A comma separated string that specifies errors which will be retried while getting the status of the request.
approvalSet	This attribute is set during the Build Approval Set step, which builds this list by going through the ProvisioningPlan to build the line items that must be approved. This variable includes all ApprovalItems that are part of the request process and are updated during the AfterScript of the approval process by assimilating the decisions and comments from the Approvals copy of the ApprovalItem.
plan	A master provisioning plan object required for building transient approval set for SAP GRC response.
identityDisplayName	Display name for identity.
userGroupsMap	The map containing UserGroup data required as an input for SAP GRC User Access Web service.
customFieldsValMap	The map containing CustomFieldsVal data required as an input for SAP GRC User Access Web service.
parameterMap	A list containing Parameter data required as an input for SAP GRC User Access Web service.
requestHeaderDataMap	A map containing RequestHeaderData required as an input for SAP GRC User Access Web service.
credentialsMap	A map to store credential information which is gathered from SAP GRC application.
requestedLineItemMap	A list containing RequestedLineItem data required as an input for SAP GRC User Access Web service.
userInfoMap	A list containing UserInfo data required as an input for SAP GRC User Access Web service.
language	SAP System Language.
requestStatusMap	The request status map containing the status information of the request received from the Request Detail Web service.
connectionTimeout	The Axis2 timeout for the Web service connection timeout. This field accepts the value in minutes.
requestNumber	The request number received after successful execution of the User Access Web service. This Request number is used by Request Detail Web service for polling.
pollingInterval	The polling interval in minutes to check the status of the request.
trace	Used for debugging this workflow and when set to true, trace will be sent to stdout.
requestStubDetailsMap	Holding user information and headerinfo to generate request detail stub.

Output of SAP GRC Request Executor workflow

Field names	Description
approvalSet	This attribute is set during the Build Approval Set step, which builds this list by going through the ProvisioningPlan to build the line items that must be approved. This variable includes all ApprovalItems that are part of the request process and is updated during the AfterScript of the approval process by assimilating the decisions and comments from the Approvals copy of the ApprovalItem.
requestStatusMap	The request status map containing the status information of the request received from the Request Detail Web service.
auditLog	Audit Log for the particular request.

Importing SAP GRC Application Rule

The IdentityIQ Rule is required for populating IdentityIQ SAP application configuration parameter with SAP GRC Connector name.

Perform the following steps to import and execute the SAP GRC application rule:

1. Create `sapGrcApplications.csv` file which contains the following columns separated by comma:

   - IdentityIQ Application name
   - Respective SAP GRC Server side Connector name

     For example:

     SAPAPPLICATION1, SAPGRCCONNECTOR

     > **Note:** **Comments can be provided in the** `sapGrcApplications.csv` **file using # symbol at the beginning of the line. For example column headers.**

     > **Note:** **If second column name is not provided than IdentityIQ application name would be treated as SAP GRC Connector name.**

2. Create `sapGrcRuleParameters.xml` file which will contain the following map of arguments that are required to pass externally to the rule:

   - **path**: path of the `sapGrcApplications.csv` file
   - **separator**: separator used in `sapGrcApplications.csv` file, to separate the IdentityIQ application name and respective SAP GRC Server side Connector name.

     For example,

     ```
     <Map>
         <entry key='path' value='E://SAPGRCApplications.csv'/>
         <entry key='separator' value=','/>
     </Map>
     ```

3. Import `sapGrcApplicationsRule.xml` IdentityIQ Rule which populates IdentityIQ SAP application configuration parameter with SAP GRC Connector name.

   - The `sapGrcApplicationsRule.xml` file is present in `WEB-INF/config` folder.
   - From console run the following commands:

     ```
     import sapGrcApplicationsRule.xml

     rule "Mapping GRC Connector Name to SAP based Application"   <path of
     sapGrcRuleParameters.xml file>
     ```

For example, rule "Mapping GRC Connector Name to SAP based application"
"E://sapGreRuleParameters.xml"

Following figure displays the output of the above performed steps:

Logging for the Rule -

Enter the following line to set logging for the rule in log4j.properties file:

```
log4j.logger.SAPGRC.sapGrcApplicationsRule=debug
```

Viewing the reports

SAP GRC related transaction in access request status report will be displayed against Comments with **External GRC System-SAP GRC** string, which is located at **Intelligence => Reports => Category:Lifecycle Manager Reports => Access Request Status Report**.

Upgrade considerations

Note: **Any customizations done on SAP GRC before upgrading to IdentityIQ version 7.0 Patch 2 will not be reflected in SAP GRC after upgrading to IdentityIQ version 7.0 Patch 2.**

To support proactive check and SAP CUA integration in SAP GRC, import
Workflow_SAPGRC_Integration.xml file while upgrading to IdentityIQ version 7.0 Patch 2.

Note: **Perform Step 2. of "Importing integration workflows" section.**

Additional information

This section describes the additional information related to the SAP GRC integration.

Creating a RFC Connection on SAP GRC system

The following steps are used to create RFC connection which can be used as Request Initiation System to indicate IdentityIQ connection virtually at SAP GRC server:

1. Execute **TCODE SM59** or navigate to **SPRO ==> SAP Reference IMG ==>Governance Risk and Compliance ==> Common Component Settings ==> Integration Framework ==> Create Connectors** and execute it. The Configuration of RFC Connections page is displayed.

2. Navigate to **ABAP Connections** and click on the create icon.

3. Give a name to the RFC Destination in new screen and provide connection type as 3 means ABAP connection.

4. Enter the details on **Technical Settings**, **Logon & Security** tabs accordingly and click on Test connection and save the changes.

5. Navigate to **SPRO ==> SAP Reference IMG ==> Governance Risk and Compliance ==> Common Component Settings ==> Integration Framework ==> Maintain Connectors and Connection Types** and execute it.

6. In new screen click on **Define Connectors**. In right hand section with name **Connection type definition** click on **SAP** and double click on **Define Connectors** again in the left hand side section.

7. In new screen click on **New Entries**.

8. Select the **Target Connector** from the drop down box which is defined in Step 3. The name of the **Source Connector** and **Logical Port** must be same as that of **Target Connector**. Select the **Connection Type** as **SAP**.

9. Select the created new entry and click on **Define Connector Groups** in the left hand section. Click on **New Entries**.

10. Provide name for new connector group in column **Conn Group** in the screen at left side.

11. Provide any **Connector Group Text** and **Con. Type** as **SAP** and save it.

12. Select the created Define Connector Group and double click on **Assign Connector Group to Group Types** in the left side of the same screen.

13. In the new screen, click on **New Entries** and provide the **Connector Group Type** as **Logical Group** on the right hand side of the screen.

14. Select the created **Connector Group Type** and double click on **Assign Connectors to Connector Group** section in left side of the screen.

15. In the new screen, click on **New Entries** and provide same name which was defined in Step 3. under column **Target Connector** in right side screen. Provide **Connection Type** as **SAP** in the same screen and save it.

16. Navigate to **SPRO ==> SAP Reference IMG ==> Governance Risk and Compliance ==> Common Component Settings ==> Integration Framework ==> Maintain Connection Setting** and execute it.

17. A new window (Determine Work Area Entry) will be displayed. In this window select the **Integration Scenario** as **Auth** and click on **Continue (Enter)**.

18. Select Sub-Scenario as **AUTH** and double click on **Scenario-Connector Link** in the left hand side screen.

19. Click on **New Entries**. In the new screen on right side, select **Target Connector** name which is same as that mentioned in Step 3.

20. In Same Screen, select **Conn. Type** as **SAP**.

21. Repeat Step 16. to Step 20. for selecting the different Integration Scenario types as **PROV**, **ROLMG**, **SUPMG**.

22. Navigate to **SPRO ==> SAP Reference IMG ==> Governance Risk and Compliance ==> Access Control ==> Maintain Connection Settings** and execute it.

23. Select **Maintain Connector Settings** select **New Entries**.

24. In right hand side screen select **Target Connector** as the name defined in Step 3. and select **App Type** as **1**. Select the Environment as required and **PATH ID** as **B012**.

25. Navigate to **SPRO ==> SAP Reference IMG ==> Governance Risk and Compliance ==> Access Control ==> Maintain Mapping for Actions and Connector Groups** and execute it.

26. Select **Maintain Connector Group Status** and click on **New Entries** in left side screen.

27. In new screen in right side provide **Conn. Group** as the same name defined in Step 10. Select **Appl Type** as **001** and enable the Active check box for the respective **Conn.Group**.

28. In left hand screen double click on **Assign Default Connector To Connector Group** and click on **New Entries**.

29. Select **Conn.Group** as defined in Step 10. Select the **Target Connector** as defined in Step 3. Enable the Default check box.

> Note: **Perform the above step for all the actions and save it.**

30. To verify whether connector is added successfully or not, navigate to **SPRO ==> SAP Reference IMG ==> Governance Risk and Compliance ==> Access Control ==> Synchronization Job ==> Authorization Sync** and see whether this new connector is listed in the drop-down of connector or not.

Configuring cross system on SAP GRC

SAP GRC integration can be used to verify the risk for the request placed for cross system configuration containing SAP Role for multiple SAP Direct applications.

Perform the following steps to configure cross system on SAP GRC:

1. Create connectors for all the SAP Direct applications as mentioned in the "Creating a RFC Connection on SAP GRC system" on page 145.

2. Navigate to **SPRO ==> SAP Reference IMG ==> Governance Risk and Compliance ==> Common Component Settings ==> Integration Framework ==> Maintain Connector and Connection Types** and execute it.

3. In new screen click on **Define Connectors.** In right hand section with name Connection type definition click on **SAP**.

4. In new screen click on **New Entries** and enter the value of **Conn.Group** as **CROSS_SYST** and **Connector Group Text** as **Cross_System_Group**. Save it.

5. Select **Cross System** and on the left hand side double click **Assign Connector Group to Group Types**.

6. Click on **New Entries** and in the New screen select **Connector Group Type** as **Cross_System_Group**.

7. Select **CROSS_SYST** group and double click on **Assign Connectors To Connector Groups**.

8. Click on **New Entries** and add the connector Names configured in Step 1.

Troubleshooting

1 - IdentityIQ Rule displays an error message when '&' is used as a separator

The IdentityIQ Rule displays the following error message when '&' is used as a separator in `.csv` file:

```
java.lang.RuntimeException
```

Resolution: Add the separator in the `sapGrcRuleParameters.xml` file in the following format:

```
<Map>
    <entry key='path' value='<path of .csv file>'>
    <entry key='separator' value='&amp'/>
</Map>
```

3 - After IdentityIQ is upgraded, when performing Provisioning operation, an error message is displayed.

The following error message is displayed when performing the provisioning operation after upgrading IdentityIQ to version 7.0 patch 2:

```
An unexpected error occurred: Execution of the Access Request Web service resulted
in error. Message Type: ERROR, Message Reason: Role Type is mandatory
```

Resolution: Perform Account-Group Aggregation task.

4 - While requesting an access for an identity from IdentityIQ an error message appears.

While requesting an access for an identity from IdentityIQ, the following error message appears:

```
RABAX in SAP GRC Integration
```

Resolution: Roles which are requested, must have provisioning status set as **Production** on SAP GRC Server.

To set the status of role as **Production**, the Role maintenance quick link from the section Role Management can be used in NWBC user interface.

5 - Request gets provisioned even if there is a risk in the request.

Request gets provisioned even if there is a risk in the request as **GRAC_RISK_ANALYSIS_WOUT_NO_WS** web service was not returning an error message if correct permissions were not given to the service account.

Resolution: To resolve this issue implement the following SAP Note in the SAP GRC Server:

2187803 - GRAC_RISK_ANALYSIS_WOUT_NO_WS does not return correct error message

6 - Mitigation comments are not displayed in Access Request Status report.

When the Account name for the identity is in lower case, mitigation comments are not displayed in Access Request Status report.

Resolution: Account Name (User Name) should always be in upper case letters

7 - Unable to request SAP direct profiles through GRC Integration

While requesting the profile, the following error message is displayed:

```
An unexpected error occurred: Undefined argument: startDate : at Line: 166
```

Resolution: Implement the following SAP Note in the SAP GRC server:

2194063 - UAM: Request status IDM service doesn't return reqstatus and reqstatus_txt and request detail service doesn't return comment, approvers and correct

Appendix

This section contains information on the following:

- "A: Common Identity Management Integration Configuration" on page 153

Appendix A: Common Identity Management Integration Configuration

This appendix describes the following information.

Overview

This appendix describes configuration process for integrations with identity management (IDM) systems and the places in IdentityIQ that use the integrations. It does not describe the details of a specific integration only the general framework common to all integrations.

Creating the IntegrationConfig Object

The first step in configuring an integration is designing an instance of the IntegrationConfig object. There is currently no user interface for editing these objects, you must write them in XML and import them. The IntegrationConfig defines the following things:

- Java class that handles communication with the IDM system
- Connection parameters such as host name, user name, and password
- IdentityIQ Application object that represents the IDM system in aggregations
- List of the applications that are managed by the IDM system
- Resource and attribute name mappings
- Role synchronization style
- Methods for selecting roles to synchronize

Here is an example of `IntegrationConfig` file that has all of the options:

```
<IntegrationConfig name='Example Integration'
   executor='sailpoint.integration.ExampleIntegration'
   roleSyncStyle='it'>

<!--
  Application representing the IDM system in IIQ
-->
<ApplicationRef>
  <Reference class='Application' name='Example Integration'/>
</ApplicationRef>

<!--
  Connection parameters needed by the executor.
-->

<Attributes>
  <Map>
```

```
        <entry key='url' value='http://somehost:8080/rest/iiq'/>
        <entry key='username' value='jlarson'/>
        <entry key='password' value='1:987zxd9872970293874'/>
    </Map>
</Attributes>

<!--
  Definitions of managed resources and name mappings.
-->
<ManagedResources>
 <ManagedResource name='LDAP 42'>
  <ApplicationRef>
   <Reference class='Application' name='Corporate Directory'/>
  </ApplicationRef>
  <ResourceAttributes>
   <ResourceAttribute name='memberOf' localName='groups'/>
  </ResourceAttributes>
 </ManagedResource>
</ManagedResources>

<!--
  Synchronized role list.
  In practice you will never have a SynchronizedRoles with
  the RoleSyncFilter or RoleSyncContainer elements. All three
  are shown only as an example.
-->
<SynchronizedRoles>
  <Reference class='Bundle' name='role1'/>
  <Reference class='Bundle' name='role2'/>
</SynchronizedRoles>
```

The executor attribute has the name of a class that implements the sailpoint.object.IntegrationExecutor interface. This class is conceptually similar to a Connector class in that it does the work specific to a particular integration. Each integration package will come with an example IntegrationConfig that contains the executor class name.

The roleSyncStyle attribute defines how roles are synchronized between IdentityIQ and the IDM system. The possible values are:

- **none**: roles are not synchronized
- **detectable**: detectable (IT) roles are synchronized
- **assignable**: assignable (business) roles are synchronized
- **dual**: both detectable and assignable roles are synchronized

If this attribute has no value the default is none. More information on the role synchronization process is found in the Role Synchronization section.

ApplicationRef

Some integrations support identity aggregation. In these cases there is a **sailpoint.object.Application** object defined to represent the IDM system and an implementation of the **sailpoint.connector.Connector** interface that handles communication with the IDM system. This is normally a multiplexed connector that returns objects representing the IDM system account as well as accounts on managed resources. Links in the identity cube are created for the managed resource accounts as well as the IDM system account.

```
<ApplicationRef>
  <Reference class='Application' name='Example Integration'/>
```

```
</ApplicationRef>
```

The documentation of each integration must describe the supported configuration attributes.

The following attributes are reserved and can only be used for the purposes defined here.

- **roleSyncHistory**: list of objects containing a history of previous synchronizations
- **universalManager**: enables the integration as a manager of all applications

The **roleSyncHistory** attribute contains a list of **sailpoint.object.IntegrationConfig.RoleSyncHistory** objects that have information about roles previously synchronized with this integration. This includes the name of the role and the date it was synchronized. This list can be used by the role synchronization task to optimize communication with the IDM system by sending only the roles that have changed since the last synchronization.

The **universalManager** attribute is set to the string true to enable this integration as a manager for all IdentityIQ applications without a ManagedResources list. This can be helpful in test environments to validate deployment configuration as well as environments where all provisioning must be fulfilled by a single integration.

ManagedResources

If the integration supports provisioning, it must define a list of managed resources that corresponding to applications defined in IdentityIQ. This determines how provisioning plans created during certification or role assignment are divided and sent to each integration.

```
<!--
    Definitions of managed resources and name mappings.
  -->
  <ManagedResources>
   <ManagedResource name='LDAP 42'>
    <ApplicationRef>
      <Reference class='Application' name='Corporate Directory'/>
    </ApplicationRef>
    <ResourceAttributes>
     <ResourceAttribute name='memberOf' localName='groups'/>
    </ResourceAttributes>
   </ManagedResource>
  </ManagedResource>
```

The ManagedResources element contains a list of ManagedResource elements. A ManagedResource element must contain an ApplicationRef that defines the associated IdentityIQ application. The ManagedResource element might have an optional name attribute that defines the name of the resource within the IDM system. If the name is not specified it is assumed that the resource name is the same as the IdentityIQ application name.

The ManagedResource element might also contain a ResourceAttributes element that contains one or more ResourceAttribute elements. ResourceAttribute is used to define mappings between attribute names in the IDM system and IdentityIQ. ResourceAttribute has the following XML attributes.

- **name**: attribute name in the IDM system
- **localName**: attribute name in the IdentityIQ application schema

If a provisioning plan is sent to this integration with attributes that are not in the ResourceAttributes list it is assumed that the name in IdentityIQ is the same as the name in the IDM system.

The ResourceAttributes list does not define a filter for attributes sent to the IDM system it only defines name mappings. When an integration has an IdentityIQ application in the ManagedResource list it is assumed that all attribute requests for that application are sent to that integration. You cannot have more than one integration managing different sets of attributes for the same application.

There is a special attribute that can be defined in Attributes that declares the integration as the manager of all applications in IdentityIQ regardless of the content of the ManagedResources element. You can still use ManagedResources to define name mappings for certain applications when necessary.

SynchronizedRoles

The SynchronizedRoles element is used to define a concrete list of roles to consider when roles are synchronized. When this is included in an IntegrationConfig object it has priority over both RoleSyncFiler and RoleSyncContainer elements if they are also included.

```
<SynchronizedRoles>
  <Reference class='Bundle' name='role1'/>
  <Reference class='Bundle' name='role2'/>
</SynchronizedRoles>
```

This can be used to synchronize simple integrations with a small set of roles that does not change often. If the set of roles is large or frequently changing, it is better to use RoleSyncFilter or RoleSyncContainer.

RoleSyncFilter

The RoleSyncFilter element contains a filter that is used to identify the roles to consider for synchronization.

```
<RoleSyncFilter>
  <Filter property='syncFlag' operation='EQ' value='true'/>
</RoleSyncFilter>

<RoleSyncFilter>
  <Filter property='name' operation='LIKE' matchMode='START'
value='Sync'/>
</RoleSyncFilter>
```

Synchronization filters typically used extended role attributes. In the first example an extended attribute syncFlag must be configured and have a value of true for the role to be synchronized.

Naming conventions can also be used to identify synchronized roles. In the second example any role whose name starts with Sync is synchronized.

A RoleSyncFilter can be combined with a RoleSyncContainer. If both are specified the intersection of the two role sets is considered for synchronization.

RoleSyncContainer

The RoleSyncContainer element defines the set of roles to be considered for synchronization by identifying an inherited role.

In this example, any role that directly or indirectly inherits the role named Roles To Synchronize is synchronized. This is typically a container role that has no function other than organizing other roles.

Specifying roles with inheritance has the advantage of creating a node in the modeler tree for Roles To Synchronize that you can expand to quickly see all of the synchronized roles.

A RoleSyncContainer can be combined with a RoleSyncFilter. If both are specified the intersection of the two role sets are synchronized.

> **Note:** **If an IntegrationConfig does not have any SynchronizedRoles, RoleSyncFilter, or RoleSyncContainer elements and the roleSyncStyle element has a value other than none, it is assumed that all roles are considered for synchronization.**

Aggregation

Some integrations support feeds of identity information through the normal aggregation process. In these cases the integration package will have a SailPoint.connector.Connector implementation class and an example SailPoint.object.Application object in XML.

IDM connectors are usually multiplexed connectors that return objects representing the IDM system account as well as accounts on all managed resources.

When an aggregation application is defined a reference to it should be placed in the IntegrationConfig. This enables provisioning operations to obtain the account name in the IDM system that corresponds to an identity in IdentityIQ.

Role Synchronization

Role synchronization is performed by running the standard system task named Synchronize Roles. This task is defined in the file `tasksRunnable.xml` and is created during the normal initialization and upgrade processes.

The task normally attempts synchronization with every **IntegrationConfig** stored in the repository. The task has one hidden input argument, integrations, that can be set to a CSV of names of IntegrationConfig objects. Use this to restrict the synchronization to a particular set of integrations.

Each **IntegrationConfig** has a **roleSyncStyle** attribute that determines how roles are synchronized. If this attribute is missing or set to none, role synchronization is disabled.

When synchronization is enabled, the task first determines the set of candidate roles by evaluating the filtering options defined in the IntegrationConfig. If there is a SynchronizedRoles element it defines the concrete list of candidate roles. Otherwise the RoleSyncFilter and RoleSyncContainer are evaluated and intersected to produce the set of candidate roles.

> **Note:** Candidate roles are not necessarily the ones that are sent to the IDM system. The roles sent are further constrained by the synchronization style.

roleSyncStyle=detectable

When the synchronization style is detectable the candidate role list is filtered to contain only detectable roles as defined by the role type.

For each candidate role a simplified representation of the role is built using the SailPoint.integration.RoleDefinition class, this is called the target role.

Entitlements for the target role are extracted from the candidate in one of two ways. If the candidate role has a provisioning plan, the plan defines the resources and attribute values that are included in the target role. If the candidate role has no provisioning plan, a set of resource attributes is derived by analyzing the profile filters in the candidate role.

To give a role a provisioning plan, design an instance of the SailPoint.**object.ProvisioningPlan** as an XML and place it inside the XML for the SailPoint.object.Bundle object representing the role.

Using provisioning plans gives you more control over the contents of the target role. Profile filters might be ambiguous or result in more attribute values for the role than are necessary, but for relatively simple filters it can be easier than defining provisioning plans.

To derive target roles, first build a list of candidate profiles. If the role option or Profiles is false, than all profiles defined in the role become candidates. If the or Profiles option is true, then only the first profile in the role is a candidate. Then iterate over each filter in each candidate profile applying this algorithm:

```
if the filter is EQ or CONTAINS_ALL
```

```
  add the values for this attribute comparison to the role
if the filter is OR
  recurs for the first child filter term
if the filter is AND
  recurs for all child filter terms
```

If the role inherits other roles, the hierarchy is flattened and inherited entitlements are merged into the target role. The same process described above is applied to every role in the inheritance hierarchy.

roleSyncStyle=assignable

When the synchronization style is assignable the candidate role list is filtered to contain only assignable roles as defined by the role type.

For each candidate role, build a simplified representation of the role using the SailPoint.integration.RoleDefinition class, this is called the target role.

Entitlements for the target role are extracted from the candidate by first applying the process described in roleSyncStyle=detectable to the candidate role. This might not have any effect since assignable roles do not normally have provisioning plans or profiles.

Next the roleSyncStyle=detectable process is applied to each of the required roles referenced by the candidate role. This is typically where most of the entitlements are found.

roleSyncStyle=dual

This is a hybrid of the assignable and detectable styles used only by the IBM Tivoli Identity Manager integration.

First detectable roles are synchronized as defined in the <ZBlueXref>"roleSyncStyle=detectable" on page 157 section.

Next assignable roles are synchronized. Instead of entitlements the roles have an extended attribute containing the names of all roles that were on the required list. The integration executor might further annotate the role definition with rules for automated assignment.

Provisioning

Provisioning can be performed in several ways.
- After role assignment from the IdentityIQ identity edit page
- After role assignment from the Access Request Manager
- During certification to handle revocations and role completions
- In a background reconciliation task
- During aggregation

Both IdentityIQ and the Access Request Manager (ARM) launch workflows with provisioning being done at the end. This provides the opportunity to insert an approval step before provisioning. The default workflow for IdentityIQ identity edits is named Identity Update. By default it has no approvals but does attempt provisioning. The example workflow for ARM requests is named ARM Role Approval Example.

Certifications can do provisioning to remove entitlements and roles that were revoked as well as add missing entitlements that are necessary to satisfy a role assignment.

A reconciliation task is an instance of the Identity Refresh task template with the provisioning argument set to true. This argument is visible in the configuration page for the refresh task. Reconciliation compares the assigned

roles with the detected entitlements and automatically provisioning any missing entitlements. Entitlements might be missing due to either changes in role assignments for an identity, or changes to the definition of roles already assigned to an identity.

Reconciliation is intended to replace the IdentityIQ Provisioning. The old provisioning page was role oriented, monitored changes to roles, and sent provisioning requests for users assigned to modified roles. It did not detect changes to the assigned roles list of identities, however. The reconciliation task is identity oriented and calculates all changes necessary to make an identity's entitlements match the currently assigned roles.

Since reconciliation is now part of the core set of identity refresh options, it can also be done during aggregation. This is less common, but aggregation could change account attributes that are used by role assignment rules resulting in changes to the assigned and detected role lists. With provisioning enabled, the aggregation could trigger the provisioning of missing entitlements for the assigned roles. A common use case for this would be aggregating from an application representing a HR system with HR attributes determining assigned business roles.

Note: **Automated provisioning done by the reconciliation task or within workflows typically does not remove entitlements, it only adds missing entitlements. Removal of unnecessary entitlements is expected to be done in a certification where a user has more control. While it is possible to enable removals during automated provisioning, it is potentially dangerous and should not be done without careful consideration.**

Provisioning with Synchronized Roles

The provisioning sub-system works with the role synchronization sub-system to determine how role assignments are provisioned. If roles are not being synchronized, the raw entitlements needed by that role are compiled and sent to the integration executors. If an integration supports role synchronization, requests are compiled to the IDM system itself to add or remove native role assignments.

There might be median cases where an integration does role synchronization, but only manages a subset of the possible applications. In these cases plans are compile with both IDM system role assignments and as raw entitlements for the entitlements that are not covered by a native role assignment.

Provisioning